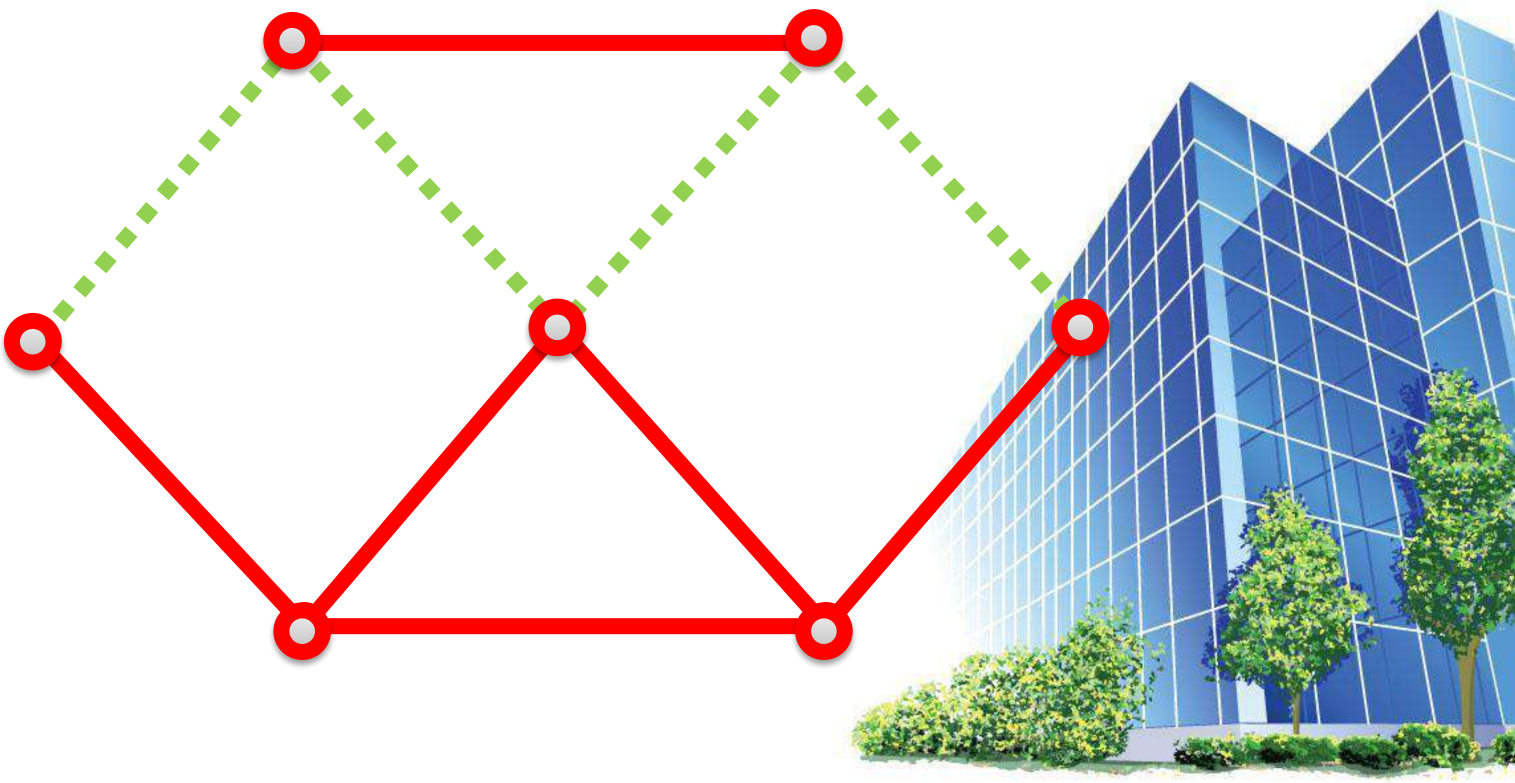
ĐẶNG NGỌC HOÀNG THÀNH

GIÁO TRÌNH

TOÁN RỜI RẠC



ĐẶNG NGỌC HOÀNG THÀNH

GIÁO TRÌNH

TOÁN RỜI RẠC

Huế, 2011

MỤC LỤC

CHƯƠNG 1. MỞ ĐẦU

1.1. Tập hợp

1.1.1. Khái niệm tập hợp

Tập hợp là một khái niệm nguyên thủy. Người ta thừa nhận khái niệm này như một lẽ tất yếu mà không đưa ra một định nghĩa cụ thể. Các đối tượng trong thế giới hợp thành một tập hợp. Tập các sinh viên trong một lớp học. Tập các số tự nhiên. Tập các đường thẳng trong mặt phẳng. Tập các quốc gia trong một châu lục. Tập các cây trong rừng. Tập các phân tử nước trong một giọt nước…. Và hà hàng sa số những ví dụ về tập hợp.

Trong tập hợp, các yếu tố bên trong nó được xem là các phần tử của tập hợp. Một tập hợp có thể không chứa phần tử nào, cũng có thể chứa hữu hạn phần tử hoặc vô hạn các phần tử.

Một tập hợp đôi khi còn được gọi tắt là tập. Ví dụ tập hợp A hay tập A.

Cho một tập hợp A, và một phần tử a của tập hợp A. Ta nói rằng, phần tử a thuộc tập hợp A. Kí hiệu

$$a \in A$$

Đọc là: phần tử A thuộc tập hợp A hoặc phần tử a thuộc tập A. Ngược lại, nếu phần tử b không phải là phần tử của tập hợp A thì ta nói rằng, phần tử b không thuộc tập hợp A và kí hiệu

$$b \notin A$$

Các cách biểu diễn tập hợp

Để biểu diễn một tập hợp, thông thường người ta sử dụng một trong hai cách sau:

a) <u>Liệt kê các phần tử của tập hợp</u>

Đối với phương pháp này, ta liệt kê tất cả hoặc một phần các phần tử trong tập hợp đó.

$A = \{0, 2, 4, 6, \dots\}$ - Tập các số tự nhiên chẵn

$A = \{a, b, c\}$ – Tập 3 kí tự a, b và c.

b) <u>Sử dụng các mô tả về tập hợp</u>

Thông thường, cách mô tả tập hợp này áp dụng cho tập có vô hạn các phần tử. Ta sẽ không liệt kê tất cả các phần tử của nó mà chỉ nêu các tính chất đặc trưng của tập hợp.

$$A = \{a \in N | a\ chia\ hết\ cho\ 2\}$$

Hay

$$A = \{a \in N | a\ \vdots\ 2\}$$

1.1.2. Tập hợp con, Tập hợp rỗng và Tập hợp bao trùm

a. Tập hợp con. Tập hợp A được gọi là con của tập hợp B, nếu mọi phần tử của tập hợp A thuộc vào tập hợp B. Kí hiệu ⊂.

$$A \subset B\ nếu\ \forall a \in A \Longrightarrow a \in B$$

Nếu tập hợp A là con của tập hợp B, thì ta cũng gọi tập hợp B là cha của tập hợp A. Khi đó, kí hiệu ⊂ sẽ được thay bằng kí hiệu ⊃.

$$(A \subset B) \Leftrightarrow (B \supset A)$$

Nếu tập hợp $A \subset B$ và $B \subset A$ thì ta nói rằng tập $A = B$.

Nếu tập hợp $A \subset B$ hoặc $A = B$ thì ta nói A là tập con hoặc bằng B và kí hiệu ⊆. Trong nhiều trường hợp, đôi khi người ta sẽ gọi $A \subseteq B$ – tập A là con của tập B và $A \subset B$ – tập A là con đúng của tập B hay nằm lọt trong B.

b. Tập hợp rỗng. Một tập hợp có thể không chứa một phần tử nào, có thể chứa hữu hạn phần tử hoặc vô hạn các phần tử. Trong trường hợp không chứa phần tử, ta gọi tập hợp này là tập hợp rỗng.

$$\Phi = \{\}$$

c. Tập hợp bao trùm. Tập hợp chứa tất cả các tập hợp khác gọi là tập hợp bao trùm (hay tập vũ trụ). Tập hợp bao trùm thường được kí hiệu là E.

Theo định nghĩa của tập hợp rỗng và tập bao trùm, ta có một số chú ý sau đây:

+ Tập hợp rỗng là con của mọi tập hợp bởi tập hợp rỗng không chứa phần tử nào.

$$\Phi \subset A, \forall A$$

+ Mọi tập hợp đều là tập con của tập bao trùm.

$$E \supset A, \forall A$$

Lực lượng của tập hợp. Số lượng các phần tử của một tập hợp được gọi là lực lượng của tập hợp. Kí hiệu $|A|$ - đọc là lực lượng của tập hợp A.

Một tập hợp có n phần tử, thì có 2^n tập hợp con.

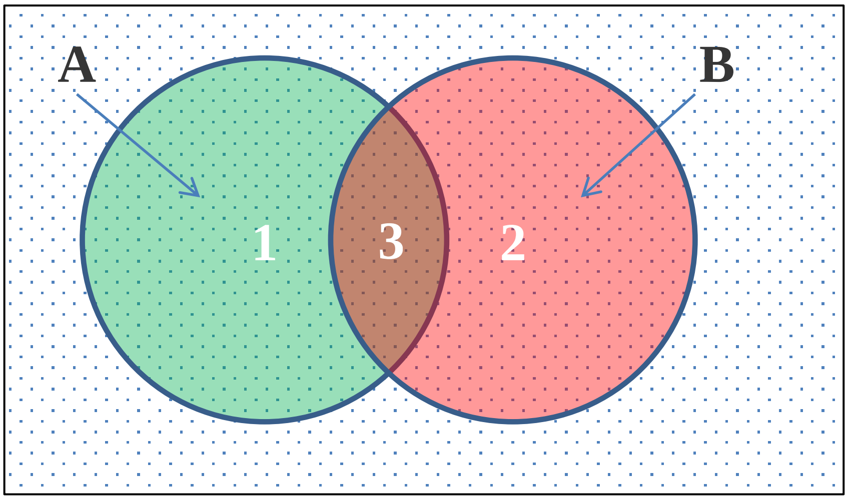
<u>Ví dụ</u>: Cho tập hợp A = {1, 2, 3}. Khi đó, sẽ có $2^3 = 8$ tập con của A. Các tập hợp này bao gồm:

$$\{\{\Phi\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$$

Tập các tập hợp con của tập hợp A được kí hiệu là $\wp(A)$.

1.1.3. Các phép toán trên tập hợp

Để minh họa cho các phép toán trên tập hợp, ta sẽ sử dụng giản đồ sau đây để minh họa (còn được gọi là giản đồ Venn).



Hình 1.1 – Minh họa các phép toán trên Tập hợp

a. Phép toán hợp. Hợp của hai tập hợp A và B là một tập hợp chứa tất cả các phần tử của tập hợp A và các phần tử của tập hợp B. Kí hiệu ∪.

$$A \cup B = \{x | x \in A \text{ hoặc } x \in B\}$$

Trong hình minh họa ở trên, hợp của hai tập hợp A và B là tập hợp chứa ba phần 1, 2 và 3.

<u>Ví dụ:</u> Giả sử ta có tập hợp A và B. Yêu cầu tìm hợp của hai tập hợp A và B.

$$A = \{1, 3\}$$

$$B = \{2,4\}$$

$$A \cup B = \{1, 2, 3, 4\}$$

Cần lưu ý rằng, tập hợp không có sự phân biệt thứ tự giữa các phần tử. Nếu tập hợp A là con của tập hợp B, thì hợp của hai tập hợp A và B chính là tập hợp B.

b. Phép toán giao. Giao của hai tập hợp A và B là một tập hợp chỉ chứa các phần tử chung của cả hai tập hợp. Kí hiệu ∩

$$A \cap B = \{x | x \in A \text{ và } x \in B\}$$

Trong hình minh họa ở trên, giao của hai tập hợp A và B là tập hợp chứa phần 3.

<u>Ví dụ:</u> Giả sử ta có tập hợp A và B. Yêu cầu tìm hợp của hai tập hợp A và B.

$$A = \{1, 2, 3\}$$

$$B = \{2,4\}$$

$$C = \{1, 3\}$$

$$A \cap B = \{2\}$$

$$B \cap C = \Phi$$

Nếu tập hợp A là con của tập hợp B, thì giao của hai tập hợp A và B chính là tập hợp A.

c. Phép toán hiệu. Hiệu của hai tập hợp A và B là một tập hợp chỉ chứa các phần tử thuộc tập hợp A mà không thuộc tập hợp B. Kí hiệu \ hoặc –. Trong một số giáo trình có sự phân biệt giữa hai kí hiệu này[1].

$$A \backslash B = \{x | x \in A \text{ và } x \notin B\}$$

Trong hình minh họa ở trên, hiệu của hai tập hợp A và B là tập hợp chứa phần 1.

<u>Ví dụ:</u> Giả sử ta có tập hợp A và B. Yêu cầu tìm hiệu của hai tập hợp A và B.

$$A = \{1, 2, 3\}$$

$$B = \{2,4\}$$

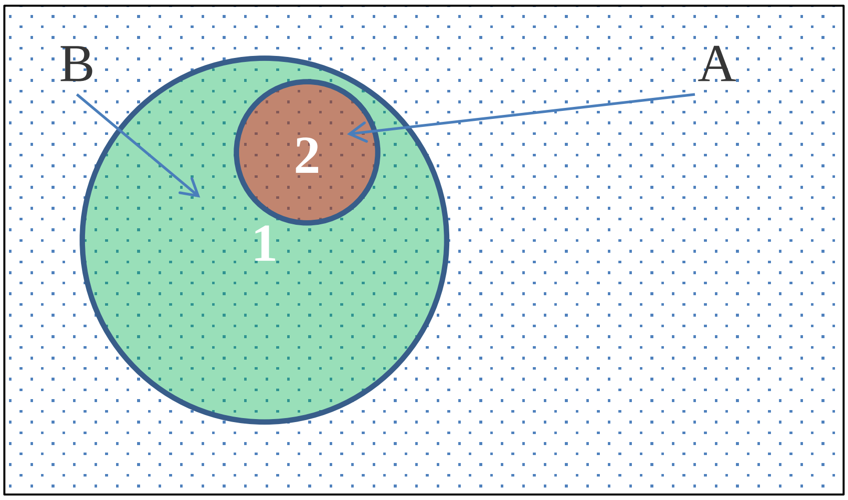
$$A \backslash B = \{1, 3\}$$

$$\mathrm{B} \backslash \mathrm{A} = \{4\}$$

Nếu tập hợp A là con của tập hợp B, thì hiệu của hai tập hợp A và B là tập hợp rỗng.

Phần bù tập hợp. Giả sử tập hợp A là con của tập hợp B và nằm lọt hẳn trong tập hợp B.

[1] Nếu A$\subset B$ thì hiệu của A và B được kí hiệu là A\B.
Nếu $A \subseteq B$ thì hiệu của A và B được kí hiệu là A-B.



Hình 1.2 – Minh họa phần bù trên Tập hợp

Khi đó, giá trị $B\backslash A$ được gọi là phần bù của tập hợp A đối với tập hợp B (hay đơn giản là phần bù của B). Kí hiệu $\overline{A}_B$. Nếu tập hợp B là tập bao trùm, ta có thể kí hiệu phần bù của tập hợp A là $\overline{A}$.

d. Phép toán hiệu đối xứng. Hiệu đối xứng của hai tập hợp A và B là một tập hợp chỉ chứa các phần tử thuộc tập hợp A mà không thuộc tập hợp B và các phần tử thuộc tập hợp B mà không thuộc tập hợp A. Kí hiệu Δ .

$$A\Delta B = (A\backslash B) \cup (B\backslash A)$$

Trong hình minh họa 1 ở trên, hiệu đối xứng của hai tập hợp A và B là tập hợp chứa phần 1 và 2.

Ví dụ: Giả sử ta có tập hợp A và B. Yêu cầu tìm hiệu đối xứng của hai tập hợp A và B.

$$A = \{1, 2, 3\}$$

$$B = \{2,4\}$$

$$A\backslash B = \{1, 3\}$$

$$\mathrm{B}\backslash\mathrm{A} = \{4\}$$

$$A\Delta B = (A\backslash B) \cup (B\backslash A) = \{1, 3, 4\}$$

Nếu tập hợp A là con của tập hợp B, thì hiệu đối xứng của hai tập hợp A và B là phần bù của tập hợp A đối với tập hợp B.

1.1.4. Các tính chất của các phép toán trên tập hợp

Chúng ta thừa nhận một số tính chất sau đây của các phép toán trên tập hợp mà không chứng minh. Việc chứng minh các tính chất này có thể thực hiện theo các luật trong logic mệnh đề.

Định lý. Giả sử A, B, C là các tập hợp. E là tập bao trùm. Khi đó, ta sẽ có các tính chất sau đây:

a) Luật giao hoán

$$A \cap B = B \cap A$$

$$A \cup B = B \cup A$$

b) Luật kết hợp

$$(A \cap B) \cap C = A \cap (B \cap C)$$

$$(A \cup B) \cup C = A \cup (B \cup C)$$

c) Luật phân phối

+ Phân phối trái

$$(A \cup B) \cap C = (A \cap C) \cup (A \cap C)$$

$$(A \cap B) \cup C = (A \cup C) \cap (A \cup C)$$

+ Phân phối phải

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

d) Luật đồng nhất

$$A \cup \Phi = A$$

$$A \cap E = A$$

e) Luật nuốt

$$A \cup E = E$$

$$A \cap \Phi = \Phi$$

f) Luật làm đầy

$$A \cup \overline{A} = E$$

$$A \cap \overline{A} = \Phi$$

g) Luật lũy đẳng

$$A \cup A = A$$

$$A \cap A = A$$

h) Luật hấp thụ

$$A \cup (A \cap B) = A$$

$$A \cap (A \cup B) = A$$

i) Luật bù

$$\bar{\bar{A}} = A$$

$$\overline{\Phi} = E$$

$$\bar{E} = \Phi$$

j) Luật De-Morgan

$$\overline{A \cap B} = \bar{A} \cup \bar{B}$$

$$\overline{A \cup B} = \bar{A} \cap \bar{B}$$

1.1.5. Khái niệm về tích Descartes

Định nghĩa. Giả sử A và B là hai tập hợp. Một tập hợp gồm các cặp (a, b) với $a \in A$ và $b \in B$, sao cho với hai cặp (a, b)=(c, d) khi và chỉ khi a=c, b=d, được gọi là tích Descartes của hai tập hợp A và B. Kí hiệu AxB hay A.B.

Ví dụ: Cho A={1, 2} và B={3, 4}. Khi đó, tích Descartes của hai tập A và B gồm các cặp sau đây:

$$A \times B = \{(1,3), (1,4), (2,3), (2,4)\}$$

Tổng quát, tích Descartes của n tập hợp $A_1, A_2, \dots, A_n$ là tập hợp gồm các cặp $(a_1, a_2, \dots, a_n)$ với $a_1 \in A_1, a_2 \in A_2, \dots., a_n \in A_n$ và $(a_1, a_2, \dots, a_n) = (b_1, b_2, \dots, b_n)$ khi và chỉ khi $a_1 = b_1, a_2 = b_2, \dots, a_n = b_n$. Kí hiệu $A_1 \times A_2 \times \dots \times A_n$.

Giả sử tập $A_1, A_2, \dots, A_n$ lần lượt có $m_1, m_2, \dots, m_n$ phần tử. Khi đó, tích Descartes của n tập hợp trên $A_1 \times A_2 \times \dots \times A_n$ sẽ có $m_1 m_2 \dots m_n$ phần tử, hay ta có thể viết

$$|A_1 \times A_2 \times \dots \times A_n| = \prod_{\mathrm{i}=1}^{\mathrm{n}} |A_i|$$

Trong đó, phép $|.|$ là kí hiệu cho lực lượng của tập hợp (tức là số phần tử của tập hợp đó).

1.2. Phép chứng minh quy nạp toán học

1.2.1. Phương pháp quy nạp toán học

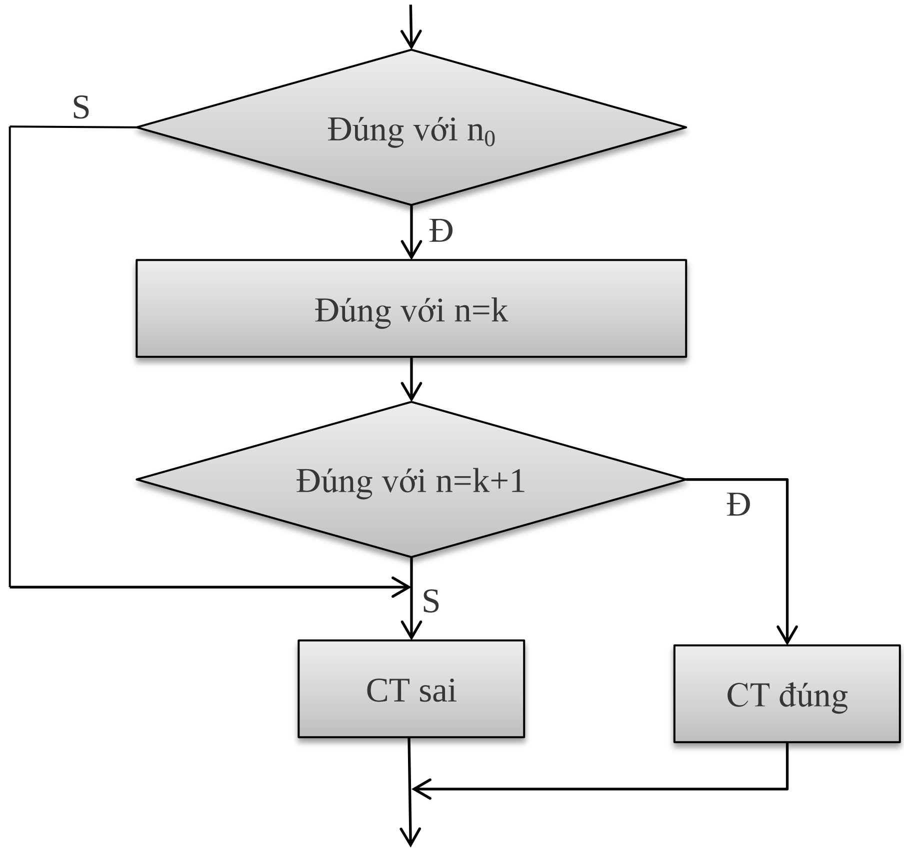
Phương pháp chứng minh quy nạp trải qua ba bước:

Bước 1. Kiểm tra công thức có đúng với trường hợp đầu tiên của chỉ số hay không. Nếu đúng ta thực hiện bước 2. Nếu sai, ta kết luận công thức sai.

Bước 2. Giả sử công thức đúng với n=k. Ta cần chứng minh công thức đúng với n=k+1. Nếu chứng minh đúng thì chuyển sang bước 3, nếu sai kết luận công thức sai.

Bước 3. Kết luận công thức đúng với mọi n.

Phép chứng minh quy nạp toán học có thể được minh họa bằng sơ đồ khối như sau:



Hình 1.3 – Sơ đồ khối của phép quy nạp toán học

Ví dụ. Chứng minh đẳng thức

$$\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$$

<u>Bước 1</u>. Với n = 0. Ta có

Vế trái VT = $\sum_{i=0}^{n} i = 0$

Vế phải VP = 0

Công thức đúng với n=0.

<u>Bước 2</u>. Giả sử công thức đúng với n=k, tức là

$$\sum_{i=0}^{k} i = \frac{k(k+1)}{2} \qquad (*)$$

Ta cần chứng minh, công thức đúng với n = k+1, tức là

$$\sum_{i=0}^{k+1} i = \frac{(k+1)(k+2)}{2}$$

Ta có

$$\sum_{i=0}^{k+1} i = \sum_{i=0}^{k} i + (k+1)$$

bởi vì

$$\sum_{i=0}^{k+1} i = \underbrace{0+1+\cdots+k}_{\sum_{i=0}^{k} i} + (k+1)$$

Kết hợp với giả thiết quy nạp (*), ta có

$$\sum_{i=0}^{k+1} i = \frac{k(k+1)}{2} + (k+1) = \frac{(k+1)(k+2)}{2}$$

<u>Bước 3</u>. Vậy, công thức đúng $\forall n \in N, n \geq 0$.

Bài tập. Chứng minh các công thức sau bằng quy nạp toán học.

$$a) \sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}, \forall n \in N$$

$$b) \sum_{i=0}^{n} i^3 = \frac{n^2(n+1)^2}{4}, \forall n \in N$$

$$c) \sum_{i=1}^{n} i.i! = (n+1)! - 1, \forall n \in N, n \geq 1$$

$$d) \sum_{i=1}^{n} \frac{i}{(i+1)!} = 1 - \frac{1}{(n+1)!}, \forall n \in N, n \geq 1$$

$e)\ 2^n > n,, \forall n \in N$

$f)\ 2^{n+2} > 2n + 5, \forall n \in N, n \geq 1$

$g)\ 3^n > n^2 + 4n + 5\ , \forall n \in N, n \geq 4$

1.2.2. Phương pháp quy nạp hoàn toàn

Trong khoa học máy tính, việc chứng minh các công thức bằng phép quy nạp nêu trên rất khó áp dụng. Nhiều bài toán trong thực tế, người ta sử dụng một phương pháp khác có thể áp dụng trên máy tính điện tử một cách dễ dàng đó là phương pháp quy nạp hoàn toàn.

Giả sử ta cần chứng minh bài toán P(n) đúng với các giá trị $n \in Z$; $n_0 \leq n \leq n_1$; $n_0, n_1 \in Z$.

Phương pháp quy nạp hoàn toàn sẽ thực hiện kiểm tra tính đúng đắn của bài toán P(n) = Q(n) với mọi giá trị n: $n_0 \leq n \leq n_1$. Khi đó, ta có thể xây dựng giải thuật như sau:

```
bool QuyNapHoanToan(int n0, int n1)
{
        for (int i=n0; i<=n1; i++)
                if (P(i)!=Q(i))
                        return false;
        return true;
}
```

Ví dụ, ta sẽ sử dụng thuật toán quy nạp hoàn toàn để kiểm tra tính đúng đắn của câu a) với giá trị $0 \leq n \leq 99999$. Chương trình sau minh họa trên ngôn ngữ C/C++.

```
#include<iostream>
#include<math.h>

using namespace std;

double P(int n)
{
        double S = 0;
        for (int i=0; i<=n; i++)
                S+=powl(i, 2);
        return S;
}
double Q(int n)
{
        return (double)(n)*(n+1)*(2*n+1)/6;
```

```
}
bool QuyNapHoanToan(int n0, int n1)
{
	for (int i=n0; i<=n1; i++)
		if (P(i)!=Q(i))
			return false;
	return true;
}
int main()
{
	if(QuyNapHoanToan(0, 9999))
		cout<<"Dang thuc dung voi moi 0<=n<=9999";
	else
		cout<<"Dang thuc sai";
	return 0;
}
```

Hãy viết giải thuật quy nạp hoàn toàn cho các bài tập còn lại.

1.2.3. Phép đệ quy và Cơ sở toán học của nó.

Trong lập trình chúng ta thường bắt gặp khái niệm đệ quy. Đệ quy đó là cách thức xác định một dãy các phần tử mà các phần tử sau được xác định thông qua phần tử trước đó. Khái niệm này tương tự như hệ thức truy hồi trong toán học. Một ví dụ điển hình đó là mô hình tính giai thừa.

$$n! = \begin{cases} 1, nếu\ n = 0 \\ n.(n-1)!, nếu\ n > 0 \end{cases}$$

Với mỗi biểu thức truy hồi, ta có thể thực hiện một phép đệ quy để xác định các phần tử của dãy. Đối với trường hợp giai thừa, ta có

- Với n=0. Ta có giai thừa của 0 là 1.

- Với n=1, ta có 1!=1.0!=1.1=1.

- Với n=2, ta có 2!=2.1!=2.1=1.

.....

- Với n=k, ta có k!=k.(k-1)!

.....

Từ cách suy diễn này, ta dễ dàng thấy được rằng, bản chất toán học của phép đệ quy đó chính là phép quy nạp toán học.

Thiết kế giải thuật đệ quy. Giả sử ta cần xây dựng giải thuật để thực hiện một thuật toán tương ứng với một giá trị n cho trước. Khi đó, ta sẽ tiến hành phân tích bài toán như sau:

Bước 1. Nếu rơi vào trường hợp suy biến, thì thực hiện trường hợp suy biến.

Bước 2. Nếu ngược lại, ta tiến hành thực hiện giải thuật gọi lại hàm đệ quy.

Ví dụ 1. Tính tổng

$$S = 1 + 2 + \cdots + n, n \in N$$

Bước 1. Trường hợp suy biến n=1 thì S=1.

Bước 2. S(n)=n+S(n-1)

Giải thuật để giải bài toàn này có thể được viết như sau:

```
long S(int n)
{
	if(n==1)
		return 1;
	else
		return n+S(n-1);
}
```

Ví dụ 2. Tìm kiếm một phần tử có giá trị x trong danh sách liên kết đơn.

Bước 1. Trường hợp suy biến khi head->data == x.

Bước 2. Nếu ngược lại, ta chuyển sang tìm kiếm ở head->next.

Giải thuật để giải bài toàn này có thể được viết như sau:

```
struct Link
{
	int data;
	Link* next;
};
Link* head;

Link* search(int x, Link* head)
{
	if(head->data==x)
		return head;
	else
		return search(int x, head->next);
}
```

Đệ quy có một ứng dụng to lớn trong khoa học máy tính. Nhiều bài toán khi thiết kế giải thuật đệ quy đơn giản hơn rất nhiều so với việc tìm kiếm một giải thuật khác. Dù

việc xây dựng một giải thuật lặp để khử đệ quy là hoàn toàn có thể thực hiện được. Trong nội dung của giáo trình này, ta sẽ tiến hành nghiên cứu một ứng dụng quan trọng của đệ quy áp dụng cho các bài toán đếm, đó là hệ thức truy hồi.

Tuy nhiên, chúng ta sẽ nghiên cứu cách giải một hệ thức truy hồi thay vì viết giải thuật đệ quy để giải nó. Việc giải hệ thức truy hồi sẽ được thảo luận trong trong phần tiếp theo của bài toán đếm.

1.3. Sơ lược về tổ hợp

1.3.1. Các quy tắc tính toán

1.3.1.1. Quy tắc cộng

Quy tắc cộng. Giả sử cần thực hiện n công việc. Các công việc thực hiện độc lập không ảnh hưởng lẫn nhau. Để thực hiện công việc thứ i, i=1, 2,…, n ta có a_i cách. Khi đó, để thực hiện một công việc trong n công việc ở trên, ta sẽ có $a_1 + a_{2+} \dots + a_n$ cách.

Dấu hiệu nhận biết. Để nhận biết một bài toán áp dụng quy tắc cộng, ta sẽ xem xét trong số các công việc thực hiện nó có quan hệ với nhau hay không. Nếu chúng hoàn toàn độc lập thì ta áp dụng quy tắc cộng.

Các ví dụ.

Bài 1. Cho hai danh sách liên kết đơn A và B. Danh sách A có n node, danh sách B có m node. Hỏi khi ghép hai danh sách trên lại với nhau thì danh sách mới thu được có bao nhiêu node.

Giải. Việc ghép nối hai danh sách liên kết đơn không làm thay đổi số node của mỗi danh sách. Danh sách mới thu được sẽ được khởi tạo từ các node của mỗi danh sách A và B. Tiến trình tạo danh sách ghép nối sẽ được thực thi qua hai giai đoạn: sao chép toàn bộ các node trong danh sách A và sao chép toàn bộ các node trong danh sách B. Do đó, danh sách ghép nối thu được, sẽ có n + m node.

Bài 2. Giả sử tổ bộ môn công nghệ phần mềm có 10 giảng viên, tổ kĩ thuật lập trình có 12 giảng viên. Việc chọn một giảng viên đi dự hội thảo khoa học có thể thực hiện bằng cách chọn một giảng viên bất kì trong các giảng viên của hai tổ nói trên. Hỏi có bao nhiêu cách chọn giảng viên đi dự hội thảo khoa học.

Giải. Chúng ta sẽ phân tích bài toán này để thấy rõ dấu hiệu áp dụng quy tắc cộng. Việc tiến hành chọn một giảng viên đi dự hội thảo khoa học sẽ được tiến hành theo hai công việc độc lập: hoặc chọn một giảng viên của tổ công nghệ phần mềm, hoặc chọn một giảng viên của tổ kĩ thuật lập trình. Hai công việc chọn lựa này là hoàn toàn không ảnh hưởng đến nhau. Do đó, ta sẽ áp dụng quy tắc cộng.

- Công việc chọn 1 giảng viên trong số 10 giảng viên của tổ công nghệ phần mềm sẽ có 10 cách chọn.
- Công việc chọn 1 giảng viên trong số 12 giảng viên của tổ kĩ thuật lập trình sẽ có 12 cách chọn.

Vậy sẽ có 10+12 cách chọn một giảng viên thuộc hai tổ khác nhau đi dự hội thảo khoa học.

Quy tắc cộng liên hệ với phép hợp trong lý thuyết tập hợp. Nếu ta gọi A là tập hợp các giảng viên của tổ bộ môn công nghệ phần mềm, B là tập hợp các giảng viên của tổ bộ môn kĩ thuật lập trình, thì khi đó, công việc chọn của chúng ta sẽ chính là lực lượng của tập $A \cup B$. Ta có $|A \cup B| = |A| + |B|$.

Bài 3. Môn Toán rời rạc của khoa Công nghệ thông tin được giảng dạy cho 5 lớp và được 5 giảng viên khác nhau đảm nhận. Mỗi giảng viên ra hai đề thi và nộp cho Tổ khảo thí và Đảm bảo chất lượng giáo dục của trường. Hỏi có bao nhiêu cách chọn ra một đề thi để tổ chức thi chung cho cả 5 lớp nói trên.

Giải. Hoàn toàn tương tự ở trên, việc tiến hành chọn một đề trong số hai đề thi của mỗi giảng viên là độc lập. Do đó, ta áp dụng quy tắc cộng. Để chọn một đề thi trong hai đề thi của mỗi giảng viên có 2 cách. Vì vậy, ta có 2 + 2 + 2 + 2 + 2 = 10 cách.

Bài 4. Cho tập hợp A có 3 phần tử. Hỏi có bao nhiêu tập con của tập A.

Giải. Các phần tử trong tập hợp là khác nhau, không phân biệt thứ tự. Các tập con của tập A bao gồm:

- Tập rỗng.
- Tập có 1 phần tử.
- Tập có hai phần tử.

- Tập có 3 phần tử.

Các tập hợp này hoàn toàn khác nhau (điều kiện cần để hai tập hợp bằng nhau là lực lượng của chúng phải bằng nhau). Do đó, ta áp dụng quy tắc cộng.

- Số lượng tập rỗng: 1 tập.
- Số lượng tập 1 phần tử: 3 tập.
- Số lượng tập 2 phần tử: 3 tập.
- Số lượng tập 3 phần tử: 1 tập.

Vậy có 1+3+3+1 = 8 tập con của tập A.

1.3.1.2. Quy tắc nhân

Quy tắc nhân. Giả sử cần thực hiện một công việc. Công việc này có thể chia ra làm n bước. Để thực hiện công việc ở bước thứ i, i=1, 2,..., n ta có a_i cách. Khi đó, để thực hiện công việc ở trên, ta sẽ có $a_1 . a_2 \dots a_n$ cách.

Dấu hiệu nhận biết. Để nhận biết một bài toán áp dụng quy tắc nhân, ta sẽ xem xét trong số các công việc thực hiện ở mỗi bước có quan hệ với nhau hay không. Nếu chúng ảnh hưởng lẫn nhau (hoặc mỗi cách lựa chọn khác nhau khi mỗi bước lựa chọn là khác nhau – tương ứng với tích Descartes) thì ta áp dụng quy tắc nhân.

Các ví dụ.

Bài 1. Trang phục của một sinh viên khi đi picnic bao gồm: áo sơ mi, quần âu, giầy và mũ. Giả sử rằng, Nam có 3 áo sơ mi, 3 quần âu, 2 đôi giầy và 2 chiếc mũ. Hỏi có bao nhiêu trang phục mà Nam có thể mặc khi đi picnic.

Giải. Hai bộ trang phục được gọi là khác nhau, khi có ít nhất một thành phần trong trang phục đó là khác nhau. Việc tiến hành chọn các thành phần trong trang phục là có quan hệ với nhau. Việc chọn 1 quần âu, sau đó kết hợp với áo sơ mi khác nhau, giầy khác nhau và mũ khác nhau cũng tạo thành một trang phục mới. Nam không thể mặc thiếu một thành phần nào trong số các thành phần nêu trên, bởi lẽ khi đó, Nam không mặc một trang phục hoàn chỉnh để tham gia đi picnic. Điều này có nghĩa là chúng ta áp dụng quy tắc nhân. Việc chọn trang phục sẽ tiến hành theo 4 bước:

- Chọn áo sơ mi – có 3 cách;
- Chọn quần âu – có 3 cách;

- Chọn giầy – có 2 cách;
- Chọn mũ – có 2 cách.

Vậy theo quy tắc nhân, ta có 3.3.2.2=36 bộ trang phục hoàn chỉnh.

Bài 2. Giả sử trong một trường đại học, cách đánh chỉ số giảng đường được quy định như sau: phần chữ và phần số. Phần chữ bao gồm một trong các kí tự sau: A, B, C, D, E, F, X, Y; phần số bao gồm các số từ 1 cho đến 50. Hỏi có bao nhiêu cách đặt tên cho các giảng đường. Giả sử số giảng đường không giới hạn.

Giải. Việc tiến hành đặt tên cho giảng đường phải tiến hành theo hai phần: phần đặt chữ và phần đặt số. Cách đặt chữ và đặt số có quan hệ với nhau: một chữ có thể kết hợp với nhiều số để đặt tên cho các giảng đường. Hai giảng đường khác nhau khi có ít nhất là phần chữ hoặc phần số là khác nhau. Do đó, ta áp dụng quy tắc nhân. Số cách đặt chữ - có 8 cách. Số cách đặt số - có 50 cách. Vậy có 8.50=400 cách đặt tên cho các giảng đường.

Bài 3. Giả sử trong một bảng mã có 20 kí tự. Cần tạo một mật khẩu có độ dài 5. Mật khẩu không phân biệt chữ hoa và chữ thường, không có một sự ràng buộc nào. Hỏi có bao nhiêu mật khẩu có thể chọn.

Giải. Việc chọn mật khẩu sẽ được tiến hành theo 5 bước – tương ứng với 5 kí tự trong mật khẩu. Chọn kí tự đầu tiên trong dãy mật khẩu – có 20 cách. Vì mật khẩu có thể chứa các kí tự trùng lặp, do đó trong các kí tự tiếp theo của mật khẩu vẫn có 20 cách chọn lựa cho mỗi vị trí. Vậy, theo quy tắc nhân, ta có $20.20.20.20.20=32.10^5$ mật khẩu được tạo thành.

1.3.2. Các cấu hình tổ hợp

1.3.2.1. Hoán vị không lặp và Hoán vị vòng

a. Hoán vị. Hoán vị không lặp (hay chính xác hơn là hoán vị không lặp tuyến tính, hay gọi tắt là hoán vị) của n phần tử khác nhau là số cách sắp xếp có thứ tự tuyến tính của n phần tử đó.

Giả sử, ta có n phần tử cần sắp xếp vào n vị trí thẳng hàng. Để tiến hành công việc sắp xếp, ta sẽ lần lượt sắp xếp các phần tử một vào các vị trí thứ nhất, thứ hai,…

- Chọn phần tử đầu tiên và xếp vào vị trí đầu tiên: có n cách.
- Chọn phần tử thứ hai và xếp vào vị trí thứ hai: có n-1 cách (vì không cho phép chọn lại phần tử trước đó).

….

- Chọn phần tử thứ n và xếp vào vị trí cuối cùng: có 1 cách.

Các công việc chọn lựa này có quan hệ với nhau: hai cách sắp xếp là khác nhau khi hai phần tử cùng chỉ số là khác nhau. Vậy theo quy tắc nhân, ta có n.(n-1)…1=n! cách sắp xếp.

Bài toán hoán vị của n phần tử có n! cách sắp xếp.

Công thức tính số hoán vị. Hoán vị thường được kí hiệu là P_n,

$$P_n = n!$$

b. Hoán vị vòng. Hoán vị vòng (hay chính xác hơn là hoán vị vòng không lặp) của n phần tử khác nhau là trường hợp khi ta hoán vị n phần tử đó trên một đường tròn.

Giả sử trên một đường tròn có n vị trí, ta cần xếp n phần tử khác nhau này lên các vị trí trên đường tròn. Khi đó, với phần tử đầu tiên ta có thể đặt nó vào mộ vị trí bất kì trên đường tròn này mà không có một sự phân biệt nào. Khi đặt phần tử thứ 2 vào khi đó mới có sự phân biệt – vị trí đặt phần tử thứ hai liền kề với phần tử thứ nhất là hoàn toàn khác biệt với việc đặt phần tử thứ hai cách phần tử thứ nhất một khoảng trống. Kể từ phần tử thứ hai này, mới có sự khác biệt. Các phần tử tiếp theo cũng tương tự như trường hợp phần tử thứ hai. Do đó, về bản chất, thì hoán vị vòng của n phần tử có thể quy về hoán vị tuyến tính của n-1 phần tử. Nghĩa là bài toán hoán vị vòng của n phần tử sẽ có (n-1)! cách sắp xếp.

Công thức tính số hoán vị vòng. Hoán vị vòng thường được kí hiệu là $\widetilde{P_n}$.

$$\widetilde{P_n} = P_{n-1} = (n-1)!$$

Chú ý. Cần lưu ý đến một số điểm sau đây khi sử dụng cấu hình hoán vị:

- Khi một bài toán yêu cầu tìm số cách sắp xếp của n phần tử vào n vị trí hay tính số cách hoán đổi vị trí của n cặp phần tử nào đó thì ta sẽ chọn cấu hình hoán vị. Nếu vị

trí sắp xếp trên đường thẳng, ta áp dụng hoán vị tuyến tính. Nếu sắp xếp trên đường tròn, ta áp dụng hoán vị vòng.

- Các cấu hình hoán vị nêu trên chỉ áp dụng cho các phần tử khác nhau. Trường hợp các phần tử có thể trùng lặp sẽ liên quan đến một cấu hình tổ hợp khác mà chúng ta sẽ khảo sát sau.

Các ví dụ.

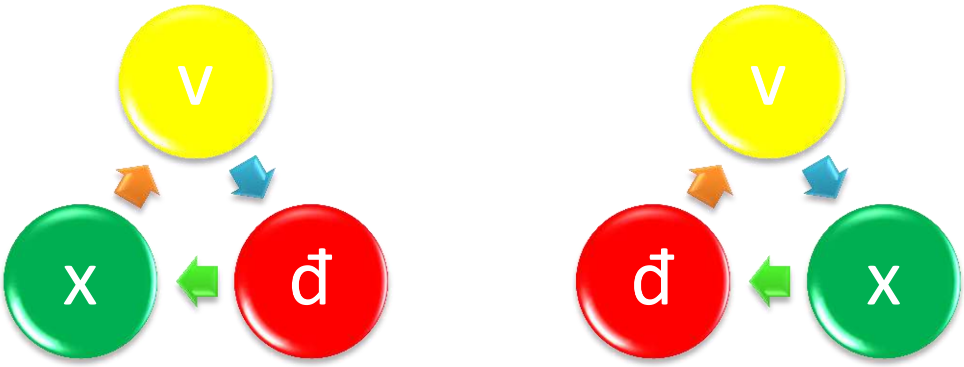
Bài 1. Có ba quả bóng xanh, đỏ và vàng. Có ba sọt được đặt theo thứ tự thẳng hàng. Hỏi có bao nhiêu cách xếp ba quả bóng trên vào ba sọt.

Giải. Rõ ràng đây là bài toán sắp xếp ba vật vào ba vị trí thẳng hàng, nên ta sẽ áp dụng cấu hình hoán vị. Nghĩa là có 3!=6 cách sắp xếp. Các cách sắp xếp đó là (xanh, đỏ, vàng), (xanh, vàng, đỏ), (đỏ, xanh, vàng), (đỏ, vàng, xanh), (vàng, xanh, đỏ) và (vàng, đỏ, xanh).

1.	**x**	**đ**	**v**
2.	**x**	**v**	**đ**
3.	**đ**	**x**	**v**
4.	**đ**	**v**	**x**
5.	**v**	**x**	**đ**
6.	**v**	**đ**	**x**

Bài 2. Nếu ba sọt trên được xếp thành vòng tròn, thì có bao nhiêu cách sắp xếp.

Giải. Rõ ràng đây là bài toán hoán vị vòng, nghĩa là có (3-1)!=2 cách.



Các cách đó là (vàng, đỏ, xanh) và (vàng, xanh, đỏ). Một số cách khác trong hoán vị tuyến tính trở nên trùng lặp trong hoán vị vòng.

Bài 3. Có bao nhiêu cách sắp xếp 5 vị đại biểu của các quốc gia khác nhau vào một hội nghị bàn tròn. Biết rằng số ghế tương ứng với số đại biểu.

Giải. Hoán vị vòng. Số cách sắp xếp là: 4!=24 cách.

Bài 4. Trong một lớp học có 40 học sinh. Hỏi có bao nhiêu cách sắp xếp chỗ ngồi cho các học sinh. Biết rằng, bất kì hai học sinh nào cũng đồng ý ngồi cùng nhau.

Giải. Hoán vị. Số cách sắp xếp là 40! cách.

1.3.2.2. Chỉnh hợp không lặp và Chỉnh hợp lặp

a. Chỉnh hợp không lặp. Chỉnh hợp không lặp chập k (chập k có nghĩa là chọn ra k phần tử) của n phần tử là số bộ gồm có k phần tử có thứ tự khác nhau của n phần tử đó.

Giả sử ta có n phần tử. Yêu cầu tìm số bộ k phần tử có thứ tự khác nhau từ n phần tử. Đó là một bài toán đơn giản. Chúng ta dễ dàng tìm được lời giải cho bài toán này nhờ vào quy tắc nhân.

Việc chọn ra k phần tử có thứ tự, tương ứng với việc chọn ra k phần tử và sắp xếp vào k vị trí thẳng hàng. Trước tiên, ta chọn ra phần tử thứ nhất và xếp vào vị trí thứ nhất – có n cách chọn, với phần tử thứ 2, ta có n-1 cách chọn,…., với phần tử thứ k, ta có n-k+1 cách chọn. Việc tiến hành chọn ở mỗi bước có quan hệ với nhau. Theo quy tắc nhân, ta có

$$n.(n-1)\dots(n-k+1) = \frac{n.(n-1)\dots 1}{(n-k).(n-k-1)\dots 1} = \frac{n!}{(n-k)!}$$

Vậy bài toán chỉnh hợp không lặp chập k có $\frac{\boldsymbol{n}!}{(\boldsymbol{n}-\boldsymbol{k})!}$ bộ phần tử.

Công thức tính số chỉnh hợp không lặp chập k của n phần tử. Chỉnh hợp không lặp chập k của n phần tử thường được kí hiệu là A_n^k.

$$A_n^k = \frac{n!}{(n-k)!}$$

b. Chỉnh hợp lặp. Chỉnh hợp lặp chập k (chập k có nghĩa là chọn ra k phần tử) của n phần tử là số bộ gồm có k phần tử có thứ tự của n phần tử đó và các phần tử này có thể lặp lại.

Giả sử ta có n phần tử. Yêu cầu tìm số bộ k phần tử có thứ tự, và các phần tử này có thể lặp lại cũng hoàn toàn tương tự như bài toán chỉnh hợp không lặp ở trên. Chúng ta dễ dàng tìm được lời giải cho bài toán này nhờ vào quy tắc nhân.

Việc chọn ra k phần tử có thứ tự, tương ứng với việc chọn ra k phần tử và sắp xếp vào k vị trí thẳng hàng. Trước tiên, ta chọn ra phần tử thứ nhất và xếp vào vị trí thứ nhất – có n cách chọn, với phần tử thứ 2, theo giả thiết các phần tử có thể lặp lại, nên ta có thể chọn lại phần tử ban đầu, nghĩa là ta có n cách chọn,…., với phần tử thứ k, ta có cũng có n cách chọn. Việc tiến hành chọn ở mỗi bước có quan hệ với nhau. Theo quy tắc nhân, ta có

$$\underbrace{n.n\ldots n}_{k\text{ lần}} = n^k$$

Vậy bài toán chỉnh hợp lặp chập k có n^k bộ phần tử.

Công thức tính số chỉnh hợp lặp chập k của n phần tử. Chỉnh hợp lặp chập k của n phần tử thường được kí hiệu là $\dot{A}_n^k$.

$$\dot{A}_n^k = n^k$$

Chú ý. Cần lưu ý đến một số điểm sau đây khi sử dụng cấu hình chỉnh hợp.

- Chỉnh hợp được áp dụng cho bài toán có phân biệt thứ tự giữa các phần tử.
- Nếu các phần tử cho phép lặp lại thì chỉnh hợp được áp dụng là chỉnh hợp lặp chập k của n phần tử; ngược lại, nếu các phần tử khác nhau thì chỉnh hợp được sử dụng sẽ là chỉnh hợp không lặp chập k của n phần tử.

<u>Các ví dụ</u>.

Bài 1. Một đội hình thi đấu cầu lông hai người thường phân biệt vận động viên bên trái và vận động viên bên phải. Giả sử rằng trong câu lạc bộ cầu lông có 10 người và các vận động viên này có khả năng đảm nhiệm vị trí chơi trái và phải là như nhau. Hỏi có bao nhiêu cách chọn ra một cặp vận động viên để đi thi đấu.

Giải. Chúng ta cần lưu ý đến một số điểm sau đây:

- Không gian chọn của chúng ta là 10 người. Bạn có thể chọn bất kì nhưng chỉ được chọn trong số 10 vận động viên này.
- Chúng ta sẽ chọn ra 2 vận động viên để đi thi đấu, như vậy k=2.
- Mỗi cặp vận động viên được chọn là có thứ tự (bởi có phân biệt trái phải). Do đó, ta áp dụng cấu hình chỉnh hợp.
- Một vận động viên không cho phép lặp lại (bởi họ không thể phân thân thành 2).

Từ những nhận xét trên, ta sẽ chọn chỉnh hợp không lặp để áp dụng cho bài toán này với k và n như đã nêu. Vậy, ta có $A_{10}^{2} = 10.9 = 90$ cách chọn các cặp vận động viên để thi đấu.

Bài 2. Quay lại với bài tập 3 trong quy tắc nhân, ta có nhận xét sau:
- Không gian chọn các kí tự cho mật khẩu là 20 kí tự trong bảng mã, tức n=20.
- Số kí tự cần chọn là 5, tức k=5.
- Các kí tự trong mật khẩu có phân biệt thứ tự, nên ta áp dụng chỉnh hợp.
- Các kí tự có thể lặp lại, nên ta áp dụng chỉnh hợp lặp.

Kết quả hoàn toàn trùng khớp với kết quả ở trên 20^5 mật khẩu.

Bài 3. Có bao nhiêu số có 4 chữ số khác nhau được lấy trong tập {1, 2, …, 9}.
Giải. Dễ nhận thấy n=9, k=4. Các số có phân biệt thứ tự và không thể lặp lại, nên ta áp dụng chỉnh hợp không lặp. Vậy có $A_9^4 = \frac{9!}{5!} = 9.8.7.6 = 3024$ số.
Nếu bài toán không yêu cầu các chữ số khác nhau thì ta sẽ áp dụng chỉnh hợp lặp.

Bài 4. Trong một cuộc thi Olympic Toán học thế giới, có 8 thí sinh của 8 nước tham dự: Việt Nam, Nga, Nhật Bản, Hoa Kỳ, Pháp, Trung Quốc, Anh và Canada. Các quốc gia sẽ tranh nhau ba giải huy chương: vàng, bạc và đồng. Hỏi có bao nhiêu khả năng xảy ra. Giả sử rằng, không có trường hợp hai thí sinh đạt cùng thành tích.
Giải. Dễ nhận thấy n=8, k=3. Việc chọn ra 3 quốc gia là có thứ tự (vì sẽ lần lượt nhận các huy chương vàng, bạc và đồng) – nên ta sử dụng chỉnh hợp, mỗi quốc gia chỉ có

một thí sinh dự thi, do đó họ không thể nhận đồng thời 2 giải - tức không lặp. Vậy có $A_8^3 = \frac{8!}{5!} = 336$ khả năng.

1.3.2.3. Tổ hợp không lặp và Tổ hợp lặp

a. Tổ hợp không lặp. Tổ hợp không lặp chập k của n phần tử là số bộ gồm k phần tử khác nhau không phân biệt thứ tự từ n phần tử đó. Số tổ hợp không lặp chập k của n phần tử được kí hiệu là C_n^k.

Ta sẽ đối chiếu tổ hợp không lặp với chỉnh hợp không lặp. Ta nhận thấy rằng, trường hợp tổ hợp không lặp chỉ là một trường hợp riêng của chỉnh hợp không lặp. Nếu ta lấy tất cả các kết quả của chỉnh hợp không lặp và hoán đổi vị trí của các phần tử (thực hiện phép hoán vị k phần tử) thì ta sẽ thu được k! Lần số tổ hợp không lặp tương ứng. Điều đó có nghĩa là số chỉnh hợp không lặp gấp k! lần số tổ hợp không lặp.

$$A_n^k = k!.\, C_n^k$$

Từ đó, ta nhận được công thức tính số tổ hợp không lặp.

Công thức tính số tổ hợp không lặp chập k của n phần tử.

$$C_n^k = \frac{A_n^k}{k!} = \frac{n!}{k!\,(n-k)!}$$

b. Tổ hợp lặp. Tổ hợp lặp chập k của n phần là số bộ gồm có k phần tử từ n phần tử đó, không phân biệt thứ tự và các phần tử có thể lặp lại.

Ta dễ nhận thấy rằng, trường hợp tổ hợp không lặp chỉ là một trường hợp riêng của tổ hợp lặp. Nếu ta loại bỏ các trường hợp lặp trong tổ hợp lặp, thì ta thu được tổ hợp không lặp.

Ta sẽ tính số tổ hợp lặp chập k của n phần tử thông qua số tổ hợp không lặp. Giả sử ban đầu, chúng ta chọn ra 1 phần tử. Khi đó, vì nó là phần tử duy nhất nên hiển nhiên không lặp. Ta cần chọn tiếp k-1 phần tử nữa. Bởi vì k-1 phần tử này trong trường hợp tổ hợp không lặp là khác với phần tử đầu tiên, nhưng với trường hợp tổ hợp không lặp thì cho phép trùng lặp. Do đó, ta có thể bổ sung vào k-1 phần tử bất kì, và tiến hành lựa chọn theo cấu hình tổ hợp không lặp (hàm ý không lặp nhưng thật chất là lặp). Nghĩa là ta sẽ có số tổ hợp không lặp chập k của n+k-1 phần tử (vì

bổ sung thêm k-1 phần tử). Nếu ta kí hiệu số tổ hợp lặp chập k của n phần tử là $\dot{C}_n^k$, ta sẽ có số tổ hợp lặp $\dot{C}_n^k = C_{n+k-1}^k$.

Công thức tính số tổ hợp lặp chập k của n phần tử.

$$\dot{C}_n^k = C_{n+k-1}^k = \frac{(n+k-1)!}{k!\,(n-1)!}$$

Chú ý. Cần lưu ý đến một số điểm sau đây khi sử dụng cấu hình tổ hợp:

- Tổ hợp được áp dụng cho bài toán không có phân biệt thứ tự giữa các phần tử.
- Nếu các phần tử cho phép lặp lại thì tổ hợp được áp dụng là tổ hợp lặp chập k của n phần tử; ngược lại, nếu các phần tử khác nhau thì tổ hợp được sử dụng sẽ là tổ hợp không lặp chập k của n phần tử.

Các ví dụ.

Bài 1. Có một giỏ hoa quả gồm có 5 loại quả là táo, lê, mận, nho và cam. Số lượng hoa quả mỗi loại là không hạn chế. Hỏi có bao nhiêu cách chọn ra 4 quả bất kì để đặt vào đĩa.

Giải. Dễ nhận thấy không gian loại quả là 5, tức n=5; số quả cần lấy ra là 4, tức k=4. Ta sẽ chọn ra 4 quả bất kì để đặt vào đĩa, nghĩa là không phân biệt thứ tự (tổ hợp) và có thể lặp lại. Vậy, ta có số cách chọn là $C_{5+4-1}^4 = \frac{8!}{4!4!} = \frac{8.7.6.5}{4.3.2.1} = 70$ cách.

Bài 2. Cho phương trình $x + y + z = 30$. Hỏi có bao nhiêu nghiệm nguyên thỏa mãn điều kiện $x \geq 5, y \geq 4, z \geq 3$.

Giải. Nếu ta chia tập các lựa chọn thành 3 nhóm 1, 2 và 3. Bài toán quy về chọn x phần tử nhóm 1, y phần tử nhóm 2 và z phần tử nhóm 3. Sao cho tổng các phần tử được chọn của cả ba nhóm là 30. Theo như điều kiện ràng buộc, ta cần chọn ra ít nhất 5 phần tử của nhóm 1 ($x \geq 5$), 4 phần tử của nhóm 2 ($y \geq 4$) và 3 phần tử của nhóm 3 ($z \geq 3$). Và tiếp tục chọn các phần tử của mỗi loại để đạt đến số 30 phần tử. Nghĩa là ta cần tiếp tục chọn thêm 30-5-4-3=18 phần tử, tức k=18, ta cần chọn các phần tử trong 3 nhóm, tức n=3. Việc tiến hành chọn các phần tử ở mỗi nhóm là không phân biệt thứ tự (bạn có thể chọn nhóm nào trước cũng đều thỏa mãn) và các phần tử có thể lặp lại. Vì vậy, ta sử dụng cấu hình tổ hợp lặp chập k của n phần tử, tức là $C_{18+3-1}^{18} = \frac{20!}{2!18!} = \frac{20.19}{2} = 190$ nghiệm thỏa mãn.

Bài 3. Có bao nhiêu cách chọn 10 đại biểu trong 100 đại biểu để bầu vào đoàn đại biểu Trưng Ương. Giả sử các vị trí trong đoàn đại biểu không phân biệt chức vụ. Mỗi đại biểu chỉ được bầu 1 lần.

Giải. Dễ dàng nhận thấy việc chọn 10 đại biểu trong 100 đại biểu sẽ tương ứng với n =100, k=10. Các vị trí trong đoàn không phân biệt thứ tự, và không được phép lặp, nên ta áp dụng tổ hợp không lặp. Ta có C_{100}^{10} cách.

1.3.2.4. Hoán vị lặp

Bài toán. Giả sử có n_1 phần tử loại 1, n_2 phần tử loại 2, …., n_k phần tử loại k. Hỏi có bao nhiêu cách sắp xếp tất cả các phần tử trên (sắp xếp tuyến tính).

Giải. Ta dễ nhận thấy rằng bài toán này là hoán vị lặp. Ta thực hiện sắp xếp $n_1 + n_2 + \cdots + n_k$ phần tử vào $n_1 + n_2 + \cdots + n_k$ vị trí. Mỗi nhóm phần tử có thể lặp lại. Để giải bài toán hoán vị lặp dạng này, ta tiến hành phân tích và chọn lựa theo từng nhóm một.

- Chọn n_1 vị trí để đặt các phần tử của nhóm 1 từ n phần tử → có $C_n^{n_1}$ cách.
- Khi đó, sẽ còn $n - n_1$ phần tử. Chọn n_2 vị trí để đặt các phần tử nhóm 2 → có $C_{n-n_1}^{n_2}$ cách.

…

- Chọn n_k vị trí để đặt nhóm k → có C_1^1 cách (bởi vì đây là nhóm cuối cùng).

Theo quy tắc nhân ta có $C_n^{n_1}.C_{n-n_1}^{n_2} \dots C_1^1$ cách.

Ta có,

$$C_n^{n_1}.C_{n-n_1}^{n_2} \dots C_1^1$$

$$= \frac{n!}{n_1!\,(n-n_1)!}.\frac{(n-n_1)!}{n_2!\,(n-n_1-n_2)!} \dots \frac{1}{1}$$

$$= \frac{n!}{n_1!\,n_2! \dots n_k!}$$

Vậy, số hoán vị lặp trong trường hợp này là $\frac{\boldsymbol{n}!}{\boldsymbol{n_1}!\boldsymbol{n_2}!\dots\boldsymbol{n_k}!}$.

BẢNG SO SÁNH CÁC QUY TẮC TÍNH TOÁN VÀ CÁC CẤU HÌNH TỔ HỢP

Quy tắc cộng	Quy tắc nhân
Các công việc độc lập, không có quan hệ với nhau.	Các công việc có liên quan nhau.

Tính chất	Hoán vị		Chỉnh hợp		Tổ hợp	
	Tuyến tính	*Vòng*	*Lặp*	*Không lặp*	*Lặp*	*Không lặp*
Phân biệt thứ tự	+	+	+	+	-	-
Cho phép lặp	-	-	+	-	+	-
Điều kiện n=k	+	+	-	-	-	-
Công thức	$n!$	$(n-1)!$	n^k	$\frac{n!}{(n-k)!}$	$\frac{(n+k-1)!}{k!\,(n-1)!}$	$\frac{n!}{k!\,(n-k)!}$

CHƯƠNG 2. BÀI TOÁN ĐẾM

2.1. Giới thiệu bài toán

Bài toán đếm là một dạng bài toán có ứng dụng to lớn trong thực tiễn. Ngành khoa học chuyên nghiên cứu về các bài toán đếm là toán học tổ hợp – một phân ngành của toán học rời rạc. Hiện nay, tốc độ xử lý của bộ vi xử lý máy tính ngày càng tăng nhanh, nên việc giải quyết bài toán đếm đã có một công cụ hữu hiệu để giải quyết. Không phải bài toán đếm nào cũng có lời giải. Đối với những bài toán chưa tìm ra được lời giải, người ta thường nghiên cứu kĩ thuật để đếm các cấu hình nghiệm của bài toán đó. Bên cạnh đó, có những bài toán đếm có thể giải quyết được bằng cách sử dụng các cấu hình tổ hợp: hoán vị, chỉnh hợp, tổ hợp và kết hợp với các quy tắc cộng, nhân.

<u>Các ví dụ</u>

Bài 1. Một biển số xe gồm 2 phần, phần đầu gồm 2 kí tự (từ A → Z), phần sau gồm 3 chữ số (0 → 9). Hỏi có thể tạo được bao nhiêu biển số theo cách đánh như trên.

Giải. Mỗi biển số gồm 2 phần: phần chữ và phần số. Phần chữ có 26^2 khả năng, phần số có 10^3 khả năng. Theo nguyên lý nhân, số biển số xe tạo được sẽ là $26^2.10^3 = 676000$.

Bài 2. Một đội tuyển dự bị học sinh giỏi của một trường nọ gồm có 3 nhóm: Toán, Lý và Hóa. Nhóm Toán gồm có 10 học sinh, nhóm Lý gồm có 8 học sinh, nhóm Hóa gồm có 7 học sinh. Giả sử không có một học sinh nào nằm trong hai nhóm dự bị. Học lực của mỗi học sinh là như nhau. Hỏi có bao nhiêu cách chọn một đổi tuyển tham dự kì thi học sinh giỏi. Biết rằng, mỗi đội tuyển tham dự cần có 2 học sinh thi Toán, 2 học sinh thi Lý và 2 học sinh thi Hóa.

Giải. Để chọn một đội tuyển tham dự học sinh giỏi, ta cần chọn 2 học sinh trong nhóm dự bị môn Toán, 2 học sinh trong nhóm dự bị môn Lý và 2 học sinh trong nhóm dự bị môn Hóa. Nghĩa là ta sẽ có lần lượt C_{10}^2, C_8^2 và C_9^2 cách chọn 2 học sinh Toán, Lý và Hóa. Theo quy tắc nhân, ta có

$$C_{10}^2.C_8^2.C_9^2 = \frac{10.9}{2}.\frac{8.7}{2}.\frac{9.8}{2} = 45.28.36 = 45360 \text{ cách chọn.}$$

Một số bài tập tự giải.

1) Có 5 quả bóng được đánh số thứ tự 1 đến 5. Hỏi có bao nhiêu cách sắp xếp để:

a) Quả bóng 1 đứng ở chính giữa.

b) Quả bóng 1 và 2 luôn đứng cạnh nhau.

2) Một tài khoản người dùng gồm có username và password. Giả sử username có 5-8 kí tự, password có 8-10 kí tự. Các kí tự này bao gồm các chữ số từ 0-9, các chữ cái trong bảng mã tiếng anh. Biết rằng, username không phân biệt chữ hoa và chữ thường, cho phép lặp. Password có phân biệt chữ hoa và chữ thường, cho phép lặp cục bộ, nhưng không cho lặp hoàn toàn (tức không cho phép các kí tự hoàn toàn giống nhau). Hỏi có bao nhiêu tài khoản có thể tạo ra.

3) Việc xử lý thông tin trong CPU được quản lý bởi hệ điều hành và thực hiện theo nguyên tắc hàng đợi. Giả sử, hệ điều hành ấn định thời gian xử lý thông tin cho một tiến trình là 5s. Sau thời gian này, nếu tiến trình chưa thực thi xong, thì nó sẽ rời hàng đợi và quay lại cuối hàng đợi để chờ được phục vụ tiếp. Nếu một tiến trình đã được phục vụ xong nhưng thời gian của nó vẫn chưa hết, thì tiến trình tiếp theo sẽ được đưa vào phục vụ tương ứng với khoảng thời gian còn thừa đó mà không cung cấp thêm 5s tiếp theo. Giả sử, có 5 tiến trình với thời gian cần xử lý lần lượt là 23s, 35s, 22s, 43s, 12s. Hỏi có bao nhiêu cách sắp xếp các tiến trình để tối ưu số lần phục vụ.

4) Cho một đa giác lồi có n cạnh. Hỏi đa giác này có bao nhiêu đường chéo.

5) Cho hai mặt phẳng song song (P) và (Q). Trên mặt phẳng (P) cho 5 điểm trong đó, không có bất kì 3 điểm nào thẳng hàng. Trên mặt phẳng (Q) cho 4 điểm, cũng không có bất kì 3 điểm nào thẳng hàng. Hỏi có bao nhiêu tứ diện có thể tạo ra.

6) Trong siêu thị, tại gian hàng bán hoa quả có 10 vị trí đặt sọt được bố trí thành hai hàng, mỗi hàng có 5 sọt. Có 10 sọt hoa quả trong đó có 8 sọt khác loại, hai sọt còn lại cùng loại và khác với 8 sọt kia. Hỏi có bao nhiêu cách sắp xếp các sọt hoa quả vào các vị trí nói trên.

7) Nước A có 10 thành phố, nước B có 15 thành phố. Giữa các thành phố trong một nước luôn có đường giao thông nối với nhau. Giữa hai nước A và B chỉ có hai đường giao thông nối với nhau. Hỏi có bao nhiêu đường đi để đi qua tất cả các thành phố của hai nước nói trên.

2.2. Nguyên lý bù trừ

Nguyên lý bù trừ. Giả sử cho A và B là hai tập hữu hạn phần tử. Khi đó, ta có công thức sau đây:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

Trong trường hợp A và B là hai tập rời nhau, ta có $|A \cap B| = 0$, do đó, ta nhận được công thức:

$$|A \cup B| = |A| + |B|$$

Nguyên lý bù trừ dạng tổng quát. Giả sử cho $A_1, A_2, \dots, A_n$ là các tập hợp hữu hạn phần tử. Khi đó, nguyên lý quy bù trừ dạng tổng quát sẽ được cho bởi công thức sau:

$$\left|\bigcup_{i=1}^{n} A_i\right| = \sum_{i=1}^{n}|A_i| - \sum_{1\le i<j\le n}\left|A_i \cap A_j\right| + \sum_{1\le i<j<k\le n}\left|A_i \cap A_j \cap A_k\right| + \dots + (-1)^{n+1}\left|\bigcap_{i=1}^{n} A_i\right|$$

<u>Chứng minh</u>. Để chứng minh công thức này, ta sử dụng nguyên lý quy nạp toán học.

Bước 1. Dễ dàng kiểm tra thấy rằng, công thức đúng với n=1, bởi vì

$$VT = |A_1|$$

$$VP = |A_1|$$

Bước 2. Giả sử công thức đúng với $n = \xi$, tức là

$$\left|\bigcup_{i=1}^{\xi} A_i\right| = \sum_{i=1}^{\xi}|A_i| - \sum_{1\le i<j\le \xi}\left|A_i \cap A_j\right| + \sum_{1\le i<j<k\le \xi}\left|A_i \cap A_j \cap A_k\right| + \dots + (-1)^{\xi+1}\left|\bigcap_{i=1}^{\xi} A_i\right|$$

Ta cần chứng minh công thức đúng với $n = \xi + 1$, tức là

$$\left|\bigcup_{i=1}^{\xi+1} A_i\right| = \sum_{i=1}^{\xi+1}|A_i| - \sum_{1\le i<j\le \xi+1}\left|A_i \cap A_j\right| + \sum_{1\le i<j<k\le \xi+1}\left|A_i \cap A_j \cap A_k\right| + \cdots$$
$$+ (-1)^{\xi+2}\left|\bigcap_{i=1}^{\xi+1} A_i\right|$$

Ta có,

$$\left|\bigcup_{i=1}^{\xi+1} A_i\right| = \left|\bigcup_{i=1}^{\xi} A_i \cup A_{\xi+1}\right|$$

Áp dụng giả thiết quy nạp cho trường hợp hai phần tử, ta có

$$\left|\bigcup_{i=1}^{\xi+1} A_i\right| = \left|\bigcup_{i=1}^{\xi} A_i\right| + |A_{\xi+1}| - \left|\bigcup_{i=1}^{\xi} A_i \cap A_{\xi+1}\right|$$

Theo giả thiết quy nạp và áp dụng tính chất phân phối, ta nhận được

$$\left|\bigcup_{i=1}^{\xi+1} A_i\right| = \sum_{i=1}^{\xi+1} |A_i| - \sum_{1\le i<j\le \xi+1} |A_i \cap A_j| + \sum_{1\le i<j<k\le \xi+1} |A_i \cap A_j \cap A_k| + \cdots$$
$$+ (-1)^{\xi+2}\left|\bigcap_{i=1}^{\xi+1} A_i\right|$$

Có nghĩa là công thức đúng với $n = \xi + 1$.

Bước 3. Vậy công thức đúng $\forall n \in N, n \geq 1$.

Mệnh đề: Trong đoạn $[a, b]$ cho trước, có $\left\lfloor \frac{b-a+1}{n} \right\rfloor$ số chia hết cho n.

Trong đó, phép toán $\lfloor \quad \rfloor$ là phép lấy sàn của một số.

Chứng minh. Một số chia hết cho n sẽ có dạng nx, với x là một số nguyên.

Theo giả thiết $a \leq nx \leq b$, suy ra có $(b - a + 1)$ số nx.

Điều này cũng có nghĩa là sẽ có $\left\lfloor \frac{b-a+1}{n} \right\rfloor$ số x thỏa mãn.■

Ví dụ 1. Trong tập X = {1,2, ..., 100} có bao nhiêu số chia hết cho 3 hoặc cho 5 hoặc cho 7.

Giải. Gọi A_i = {x ∈ X: x chia hết cho i} với i = 3, 5, 7.

Khi đó tập $A_3 \cup A_5 \cup A_7$ là tập các số trong N chia hết cho ít nhất một trong 3 số 3, 5, 7.

Áp dụng nguyên lý bù trừ, ta có:

$$|A_3 \cup A_5 \cup A_7|$$
$$= |A_3| + |A_5| + |A_7| - |A_3 \cap A_5| - |A_5 \cap A_7| - |A_7 \cap A_3|$$
$$+ |A_3 \cap A_5 \cap A_7|$$
$$= \left\lfloor\frac{100}{3}\right\rfloor + \left\lfloor\frac{100}{5}\right\rfloor + \left\lfloor\frac{100}{7}\right\rfloor - \left\lfloor\frac{100}{3.5}\right\rfloor - \left\lfloor\frac{100}{5.7}\right\rfloor - \left\lfloor\frac{100}{7.3}\right\rfloor + \left\lfloor\frac{100}{3.5.7}\right\rfloor = 55$$

Vậy có 55 số thỏa mãn.

Ví dụ 2. Trong một hội nghị bàn tròn của công ty A, các cổ đông cần đưa ra ý kiến thông qua một điều lệ của công ty bằng một cuộc bỏ phiếu kín. Một cổ đông chỉ có

thể hoặc đồng ý, hoặc không đồng ý. Nếu cổ đông chọn cả hai phương án hoặc phiếu trắng thì xem như không hợp lệ. Biết rằng có 45 cổ đông đồng ý, 20 cổ đông không đồng ý, 50 cổ đông hoặc đồng ý hoặc không đồng ý. Hỏi có bao nhiêu phiếu không hợp lệ.

Giải. Gọi

$$A = \{Các\ cổ\ đông\ đồng\ ý\}$$

$$B = \{Các\ cổ\ đông\ không\ đồng\ ý\}$$

Theo giả thiết, ta có

$$|A| = 45, |B| = 20, |A \cup B| = 50$$

Theo nguyên lí bù trừ, ta nhận được

$$|A \cap B| = |A| + |B| - |A \cup B| = 45 + 20 - 50 = 15.$$

Vậy có 15 phiếu không hợp lệ.

2.3. Công thức truy hồi

2.3.1. Khái niệm hệ thức truy hồi

Đôi khi ta rất khó định nghĩa một đối tượng một cách tường minh. Nhưng có thể dễ dàng định nghĩa đối tượng này qua chính nó. Kỹ thuật này được gọi là đệ quy mà ta đã thảo luận ở trên, và người ta có thể dùng kỹ thuật này để giải các bài toán đếm. Kỹ thuật đệ quy còn được dùng để tìm các phần tử của một dãy số, trong đó, phần tử sau sẽ được xác định thông qua các phần tử trước đó mà ta gọi nó là hệ thức truy hồi.

Ví dụ về hệ thức truy hồi.

a. Biểu thức tính giai thừa

$$n! = \begin{cases} 1, nếu\ n = 0 \\ n.(n-1)!, nếu\ n > 0 \end{cases}$$

b. Dãy Fibonacci

$$F(n) = \begin{cases} 0, nếu\ n = 0, \\ 1, nếu\ n = 1 \\ F(n-1) + F(n-2), nếu\ n \geq 1 \end{cases}$$

c. Lãi kép (lãi suất tích lũy)

Giả sử một người gửi 10.000 đô la vào tài khoản của mình tại một ngân hàng với lãi suất kép 11% mỗi năm. Sau 30 năm anh ta có bao nhiêu tiền trong tài khoản của mình.

Gọi P_n là tổng số tiền có trong tài khoản sau n năm. Vì số tiền có trong tài khoản sau n năm bằng số có sau n – 1 năm cộng lãi suất của năm thứ n, nên ta thấy dãy $\{P_n\}$ thoả mãn hệ thức truy hồi sau:

$$P_n = P_{n-1} + 0{,}11.P_{n-1} = 1{,}11.P_{n-1}$$

với điều kiện đầu P_0 = 10.000 đô la. Từ đó suy ra $P_n = (1{,}11)^n.10.000$. Thay n = 30 cho ta P_{30} = 228922,97 đô la.

2.3.2. Giải các hệ thức truy hồi

Định nghĩa. *Một hệ thức truy hồi tuyến tính thuần nhất bậc k với hệ số hằng là hệ thức truy hồi có dạng:*

$$a_n = c_1.a_{n-1} + \cdots + c_k.a_{n-k},$$

trong đó $c_1, c_2, \ldots, c_k$ là các hằng số thực và $c_k \neq 0$.

Theo nguyên lý của quy nạp toán học thì dãy số thỏa mãn hệ thức truy hồi nêu trong định nghĩa được xác định duy nhất bằng hệ thức truy hồi này và k điều kiện đầu: $a_0 = c_0, a_1 = c_1, \ldots, a_{k-1} = c_{k-1}$.

Phương pháp cơ bản để giải hệ thức truy hồi tuyến tính thuần nhất hệ số hằng là tìm nghiệm dưới dạng $a_n = r^n$, trong đó r là hằng số. Chú ý rằng $a_n = r^n$ là nghiệm của hệ thức truy hồi

$$a_n = c_1a_{n-1} + c_2a_{n-2} + \ldots + c_ka_{n-k}$$

nếu và chỉ nếu

$$r^n = c_1r^{n-1} + c_2r^{n-2} + \ldots + c_kr^{n-k}$$

Chia hai vế của phương trình này cho r^{n-k}, ta nhận được

$$r^k - c_1r^{k-1} - c_2r^{k-2} - \ldots - c_{k-1}r - c_k = 0.$$

Phương trình này được gọi là phương trình đặc trưng của hệ thức truy hồi tuyến tính hệ số hằng. Nghiệm của nó gọi là nghiệm đặc trưng của hệ thức truy hồi.

Định lý. (Về mối quan hệ giữa nghiệm của hệ thức truy hồi thuần nhất tuyến tính hệ số hằng với nghiệm của phương trình đặc trưng của nó).

Cho $c_1, c_2, \ldots, c_k$ là các hằng số thực. Giả sử rằng phương trình đặc trưng

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \ldots - c_{k-1} r - c_k = 0.$$

có k nghiệm phân biệt $r_1, r_2, \ldots, r_k$. Khi đó dãy $\{a_n\}$ là nghiệm của hệ thức truy hồi $a_n = c_1 . a_{n-1} + \cdots + c_k . a_{n-k}$,

nếu và chỉ nếu

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n + \cdots + \alpha_k r_k^n, n = 1,2, \ldots$$

trong đó $\alpha_1, \alpha_2, \ldots, \alpha_k$ là các hằng số.

Để tìm các hằng số $\alpha_1, \alpha_2, \ldots, \alpha_n$ ta cần thế các nghiệm thu được vào các điều kiện đầu và giải nó.

Đơn giản hơn, ta sẽ nghiên cứu chi tiết về hệ thức truy hồi tuyến tính thuần nhất bậc hai với hệ số hằng một cách kĩ lưỡng. Đối với hệ thức truy hồi tổng quát, ta chỉ khảo sát một số ví dụ mà không đi sâu vào từng trường hợp cụ thể.

Định lý. (Về mối quan hệ giữa nghiệm của hệ thức truy hồi thuần nhất tuyến tính hệ số hằng <u>bậc 2</u> với nghiệm của phương trình đặc trưng của nó).

Cho c_1, c_2 là các số thực. Xét phương trình đặc trưng

$$r^2 - c_1 r - c_2 = 0$$

của hệ thức truy hồi

$$a_n = c_1 . a_{n-1} + c_2 . a_{n-2}$$

Khi đó, nếu

a) Phương trình đặc trưng có hai nghiệm thực phân biệt r_1, r_2 thì hệ thưc truy hồi có nghiệm dưới dạng:

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$$

b) Phương trình đặc trưng có nghiệm kép r_0 thì hệ thức truy hồi có nghiệm dưới dạng:

$$a_n = \alpha_1 r_0^n + \alpha_2 n r_0^n$$

c) Phương trình đặc trưng không có nghiệm thực, nhưng có hai nghiệm phức liên hợp r_1, r_2. Khi đó, hệ thức truy hồi được gọi là không giải được trên trường số thực, nhưng có thể giải được trên trường số phức. Nghiệm sẽ được biểu diễn dưới dạng:

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$$

Để giải quyết trường hợp số phức, thông thường ta biểu diễn các số phức dưới dạng lượng giác hoặc dạng lũy thừa e.

Các hệ số α_1, α_2 trong công thức ở trường hợp a) b) và c) là các hằng số.

Các hệ số này sẽ được tìm nhờ vào các điều kiện ban đầu.

Các ví dụ.

a. Giải hệ thức truy hồi của dãy Fibonacci.

$$F(n) = \begin{cases} 0, \text{nếu } n = 0, \\ 1, \text{nếu } n = 1 \\ F(n-1) + F(n-2), \text{nếu } n \geq 1 \end{cases}$$

Ta có thể biểu diễn nó dưới dạng

$$F_n = F_{n-1} + F_{n-2}, F_0 = 0, F_1 = 1$$

Phương trình đặc trưng của hệ thức này là

$$r^n - 1.r_{n-1} - 1.r^{n-2} = 0$$

Phương trình này có hai nghiệm đặc trưng là

$$r_1 = \frac{1+\sqrt{5}}{2}, r_2 = \frac{1-\sqrt{5}}{2}$$

Do đó, số hạng tổng quát của dãy Fibonacci là

$$F_n = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + \alpha_2 \left(\frac{1-\sqrt{5}}{2}\right)^n$$

Ta sẽ tìm các hệ số α_1, α_2 nhờ vào điều kiện ban đầu $F_0 = 0, F_1 = 1$. Ta có hệ phương trình sau

$$\begin{cases} \alpha_1.\frac{1+\sqrt{5}}{2} + \alpha_2.\frac{1-\sqrt{5}}{2} = 1 \\ \alpha_1 + \alpha_2 = 0 \end{cases}$$

Giải hệ này, ta nhận được

$$\alpha_1 = \frac{1}{\sqrt{5}}, \alpha_2 = -\frac{1}{\sqrt{5}}$$

Vậy số hạng tổng quát của dãy Fibonacci là

$$F_n = \frac{1}{\sqrt{5}}.\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}.\left(\frac{1-\sqrt{5}}{2}\right)^n$$

b. Tìm số hạng tổng quát của hệ thức truy hồi sau:

$$a_n = \begin{cases} a_{n-1} + 2a_{n-2} \\ a_0 = 2, a_1 = 7 \end{cases}$$

Công thức trên có thể viết lại như sau

$$a_n = a_{n-1} + 2a_{n-2}, a_0 = 2, a_1 = 7.$$

Phương trình đặc trưng của hệ thức truy hồi là

$$r^2 - r - 2 = 0$$

Phương trình đặc trưng này có hai nghiệm phân biệt $r_1 = 2, r_2 = -1$.

Theo định lý ở trên, nghiệm của hệ thức truy hồi có dạng

$$a_n = \alpha_1 2^n + \alpha_2(-1)^n$$

Theo điều kiện đầu $a_0 = 2, a_1 = 7$ ta nhận được hệ phương trình sau đây để tìm giá trị của α_1, α_2.

$$\begin{cases} a_0 = \alpha_1 + \alpha_2 = 2 \\ a_1 = \alpha_1.2 - \alpha_2 = 7 \end{cases}$$

Giải hệ phương trình này, ta nhận được

$$\begin{cases} \alpha_1 = 3 \\ \alpha_2 = -1 \end{cases}$$

Vậy nghiệm của hệ thức truy hồi là

$$a_n = 3.2^n + (-1)^{n+1}$$

c. Tìm nghiệm của hệ thức truy hồi sau:

$$a_n = \begin{cases} 6a_{n-1} - 9a_{n-2} \\ a_0 = 1, a_1 = 6 \end{cases}$$

Công thức trên có thể viết lại như sau

$$a_n = 6a_{n-1} - 9a_{n-2}, a_0 = 1, a_1 = 6$$

Phương trình đặc trưng của hệ thức truy hồi

$$r^2 - 6r + 9 = 0$$

Phương trình có nghiệm kép là $r = 3$.

Do đó, công thức nghiệm tổng quát sẽ là

$$a_n = \alpha_1.3^n + \alpha_2.n.3^n$$

Sử dụng điều kiện đầu, ta nhận được

$$\begin{cases} \alpha_1 = 1 \\ \alpha_1.3 + \alpha_2.3 = 6 \end{cases}$$

Giải hệ này, ta nhận được $\alpha_1 = 1, \alpha_2 = 1$

Vậy nghiệm của hệ thức truy hồi trên là

$$a_n = 3^n(1 + n)$$

d. Tìm nghiệm của hệ thức truy hồi sau:

$$a_n = \begin{cases} a_{n-1} - a_{n-2} \\ a_0 = 0, a_1 = 1 \end{cases}$$

Công thức trên có thể viết lại như sau

$$a_n = a_{n-1} - a_{n-2}, a_0 = 0, a_1 = 1$$

Phương trình đặc trưng của nó có dạng

$$r^2 - r + 1 = 0$$

Phương trình này có hai nghiệm phức

$$r_1 = \frac{1 + i\sqrt{3}}{2}, r_2 = \frac{1 - i\sqrt{3}}{2}$$

Công thức nghiệm tổng quát có dạng

$$a_n = \alpha_1.\left(\frac{1 + i\sqrt{3}}{2}\right)^n + \alpha_2.\left(\frac{1 - i\sqrt{3}}{2}\right)^n$$

Để thuận lợi cho việc tính toán, ta sẽ biểu diễn hai số phức này dưới dạng lượng giác.

$$|r_1| = |r_2| = \sqrt{\left(\frac{1}{2}\right)^2 + \left(\frac{\pm\sqrt{3}}{2}\right)^2} = 1$$

$$\tan(\phi_1) = \frac{\sqrt{3}/2}{1/2} = \sqrt{3} \Rightarrow \phi_1 = \frac{\pi}{3}$$

$$\tan(\phi_2) = \frac{-\sqrt{3}/2}{1/2} = -\sqrt{3} \Rightarrow \phi_1 = -\frac{\pi}{3}$$

Khi đó, nghiệm có thể biểu diễn dưới dạng

$$a_n = \alpha_1 \left[(\cos\left(\frac{\pi}{3}\right) + i.\sin\left(\frac{\pi}{3}\right))\right]^n + \alpha_2 \left[r.(\cos\left(\frac{\pi}{3}\right) - i.\sin\left(\frac{\pi}{3}\right))\right]^n$$

hay

$$a_n = \alpha_1 \left(\cos\left(n\frac{\pi}{3}\right) + i.\sin\left(n\frac{\pi}{3}\right)\right) + \alpha_2 \left(\cos\left(n\frac{\pi}{3}\right) - i.\sin\left(n\frac{\pi}{3}\right)\right)$$

hay

$$a_n = (\alpha_1 + \alpha_2).\cos\left(\frac{n\pi}{3}\right) + i.(\alpha_1 - \alpha_2).\sin\left(\frac{n\pi}{3}\right)$$

Sử dụng điều kiện đầu, ta nhận được

$$\begin{cases} a_0 = (\alpha_1 + \alpha_2).1 + 0 = 0 \\ a_1 = (\alpha_1 + \alpha_2)\frac{1}{2} + i.\frac{\sqrt{3}(\alpha_1 - \alpha_2)}{2} = 1 \end{cases}$$

Từ đó, ta nhận được

$$\alpha_1 = -\frac{\sqrt{3}}{3i}, \alpha_2 = \frac{\sqrt{3}}{3i}$$

Vậy nghiệm của công thức truy hồi là

$$a_n = -\frac{2\sqrt{3}}{3}.\sin(\frac{n\pi}{3})$$

e. Tìm nghiệm của hệ thức truy hồi sau:

$$a_n = \begin{cases} 6a_{n-1} - 11a_{n-2} + 6a_{n-3} \\ a_0 = 2, a_1 = 5, a_3 = 15 \end{cases}$$

Hệ thức truy hồi có thể được viết lại

$$a_n = 6a_{n-1} - 11a_{n-2} + 6a_{n-3}, a_0 = 2, a_1 = 5, a_3 = 15.$$

Phương trình đặc trưng của hệ thức truy hồi

$$r^3 - 6r^2 + 11r - 6 = 0$$

Phương trình đặc trưng có 3 nghiệm $r_1 = 1, r_2 = 2, r_3 = 3$.

Công thức nghiệm tổng quát

$$a_n = \alpha_1 + \alpha_2 2^n + \alpha_3 3^n$$

Sử dụng điều kiện đầu, ta nhận được hệ

$$\begin{cases} \alpha_1 + \alpha_2 + \alpha_3 = 2 \\ \alpha_1 + 2.\alpha_2 + 3.\alpha_3 = 5 \\ \alpha_1 + 4.\alpha_2 + 9.\alpha_3 = 15 \end{cases}$$

Hệ này có nghiệm $\alpha_1 = 1, \alpha_2 = -1, \alpha_3 = 2$.

Vậy, nghiệm của hệ thưc truy hồi là

$$a_n = 1 - 2^n + 2.3^n$$

2.3.3. Quy về bài toán đơn giản

Để giải quyết các bài toán đếm phức tạp, ta thường phân tích bài toán để đưa về các bài toán con đơn giản hơn. Kĩ thuật phân chia này còn được biết đến với tên gọi là chia để trị. Bởi lẽ quy tắc này được áp dụng là vì, không phải lúc nào bài toán cũng đếm cũng dễ dàng giải quyết được. Nhưng sự phân chia để nhận được các bài toán nhỏ hơn cũng đòi hỏi người phân tích phải hiểu một cách sâu sắc về cấu hình cần đếm.

Giải sử ta xét bài toán đếm phức tạp, bài toán được phân tách thành α bài toán con. Giả sử rằng, n là kích thước của bài toán đếm, và mỗi bài toán con có kích thước được giảm đi β lần, tức kích thươc của nó là $\frac{n}{\beta}$.

Gọi f là hàm số phép toán cần thiết để giải bài toán con, nó là hàm theo biến số kích thước; g là hàm các phép toán cần thiết để thực hiện việc phân chia bài toán ban đầu về các bài toán con, nó là hàm theo biến kích thước.

Khi đó, ta có

$$f(n) = \alpha . f\left(\frac{n}{\beta}\right) + g(n)$$

<u>Phát biểu</u>: số phép toán cần thiết để giải bài toán ban đầu bằng tổng số các phép toán cần thiết để giải mỗi bài toán con và số phép toán cần thiết để thực hiện phép chuyển từ bài toán ban đầu sang các bài toán con.

2.3.4. Phương pháp liệt kê

Việc giải các bài toán đếm dưới dạng tổng quát và nhận được một công thức cụ thể là một điều cực kì khó khăn. Cho đến nay, rất nhiều bài toán đếm chưa có lời giải dưới dạng một công thức cụ thể. Đối với những bài toán như thế, chúng ta cần sử dụng một phương pháp liệt kê. Dù nó không chỉ ra một kết quả cụ thể, nhưng có thể sử dụng công cụ máy tính để thực hiện phép đếm.

CHƯƠNG 3. BÀI TOÁN TỒN TẠI

3.1. Giới thiệu bài toán

Trong nhiều bài toán tổ hợp, việc chỉ ra một cấu hình thỏa mãn các tính chất cho trước là một việc làm khó khăn. Đối với bài toán dạng này, để giải quyết, chúng ta cần khảo sát sự tồn tại của nó. Chúng được gọi là bài toán tồn tại.

Một bài toán tồn tại tổ hợp được xem như giải xong, nếu hoặc chỉ ra một cách xây dựng cấu hình hoặc chứng minh chúng không tồn tại.

Bài toán 1. Bài toán về 36 sĩ quan .

Có một lần người ta triệu tập từ 6 trung đoàn, mỗi trung đoàn 6 sĩ quan thuộc 6 cấp bậc khác nhau: thiếu uý, trung uý, thượng uý, đại uý, thiếu tá, trung tá về tham gia duyệt binh ở sư đoàn bộ. Hỏi rằng, có thể xếp 36 sĩ quan này thành một đội ngũ hình vuông sao cho trong mỗi hàng ngang cũng như mỗi hàng dọc đều có đại diện của cả sáu trung đoàn và của 6 cấp bậc.

Để đơn giản ta sẽ dùng các chữ cái in hoa A, B, C, D, E, F để chỉ phiên hiệu của các trung đoàn, các chữ cái in thường a, b, c, d, e, f để chỉ cấp bậc. Bài toán này có thể tổng quát hoá nếu thay 6 bởi n. Trong trường hợp n = 4 một lời giải của bài toán 16 sĩ quan là:

Ab	Dd	Ba	Cc
Bc	Ca	Ad	Db
Cd	Bb	Dc	Aa
Da	Ac	Cb	Bd

Một lời giải với n = 5 là:

Aa	Bb	Cc	Dd	Ee
Cd	De	Bd	Ab	Bc
Eb	Ac	Bd	Ce	Da
Be	Ca	Db	Ec	Ad

Dc Ed Ae Ba Cb

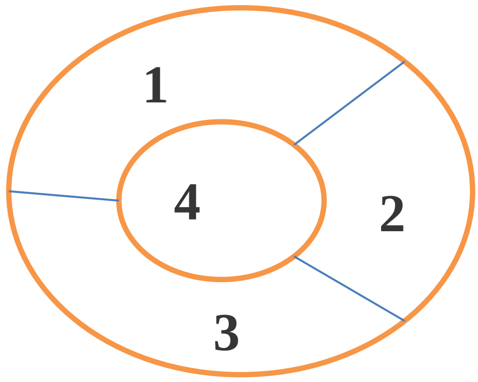
Do lời giải bài toán có thể biểu diễn bởi hai hình vuông với các chữ cái la tinh hoa và la tinh thường nên bài toán tổng quát đặt ra còn được biết với tên gọi *"hình vuông la tinh trực giao"*. Trong hai ví dụ trên ta có hình vuông la tinh trực giao cấp 4 và 5.

Euler đã mất rất nhiều công sức để tìm ra lời giải cho bài toán 36 sĩ quan thế nhưng ông đã không thành công. Vì vậy, ông giả thuyết là cách sắp xếp như vậy không tồn tại. Giả thuyết này đã được nhà toán học pháp Tarri chứng minh năm 1901 bằng cách duyệt tất cả mọi khả năng xếp. Euler căn cứ vào sự không tồn tại lời giải khi n=2 và n = 6 còn đề ra giả thuyết tổng quát hơn là không tồn tại hình vuông trực giao cấp 4n + 2. Giả thuyết này đã tồn tại hai thế kỷ, mãi đến năm 1960 ba nhà toán học Mỹ là Bore, Parker, Srikanda mới chỉ ra được một lời giải với n = 10 và sau đó chỉ ra phương pháp xây dựng hình vuông trực giao cho mọi n = 4k + 2 với k > 1.

Tưởng chừng bài toán chỉ mang ý nghĩa thử thách trí tuệ con người thuần tuý như một bài toán đố. Nhưng gần đây, người ta phát hiện những ứng dụng quan trọng của vấn đề trên vào qui hoạch, thực nghiệm và hình học xạ ảnh.

Bài toán 2. Bài toán 4 màu.

Có nhiều bài toán mà nội dung của nó có thể giải thích được với bất kỳ ai, lời giải của nó ai cũng cố gắng thử tìm nhưng khó có thể tìm được. Ngoài định lý Fermat thì bài toán bốn màu cũng là một bài toán như vậy. Bài toán có thể được phát biểu như sau: Chứng minh rằng mọi bản đồ đều có thể tô bằng 4 màu sao cho không có hai nước láng giềng nào lại bị tô bởi cùng một màu. Trong đó, mỗi nước trên bản đồ được coi là một vùng liên thông, hai nước được gọi là láng giềng nếu chúng có chung đường biên giới là một đường liên tục.



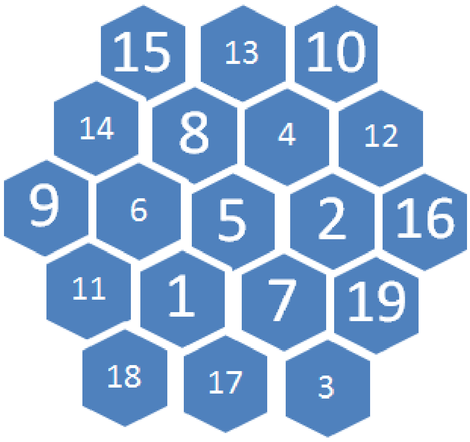
Hình 3.1 – Bài toán 4 màu

Con số bốn màu không phải là ngẫu nhiên. Người ta đã chứng minh được rằng mọi bản đồ đều được tô bởi số màu lớn hơn 4, còn với số màu ít hơn 4 thì không thể tô được, chẳng hạn bản đồ gồm 4 nước như trên hình 2.2 không thể tô được với số màu ít hơn 4.

Bài toán này xuất hiện vào những năm 1850 từ một lái buôn người Anh là Gazri khi tô bản đồ hành chính nước Anh đã cố gắng chứng minh rằng nó có thể tô bằng bốn màu. Sau đó, năm 1852, ông đã viết thư cho De Morgan để thông báo về giả thuyết này. Năm 1878, Keli trong một bài báo đăng ở tuyển tập các công trình nghiên cứu của Hội toán học Anh có hỏi rằng bài toán này đã được giải quyết hay chưa? Từ đó bài toán trở nên nổi tiếng, trong suốt hơn một thế kỷ qua, nhiều nhà toán học đã cố gắng chứng minh giả thuyết này. Tuy vậy, mãi tới năm 1976 hai nhà toán học Mỹ là K. Appel và W. Haken mới chứng minh được nó nhờ máy tính điện tử.

Bài toán 3. Hình lục giác thần bí.

Năm 1890 Clifford Adams đề ra bài toán hình lục giác thần bí sau: trên 19 ô lục giác hãy điền các số từ 1 đến 19 sao cho tổng theo 6 hướng của lục giác là bằng nhau (và đều bằng 38). Sau 47 năm trời kiên nhẫn cuối cùng Adams cũng đã tìm được lời giải. Sau đó vì sơ ý đánh mất bản thảo ông đã tốn thêm 5 năm để khôi phục lại. Năm 1962 Adams đã công bố lời giải đó. Nhưng thật không thể ngờ được đó là lời giải duy nhất.



Hình 3.2 – Bài toán hình lục giác thần bí

3.2. Phương pháp phản chứng

Ý tưởng: Phương pháp chứng minh phản chứng dựa trên hệ luật logic sau:

$$(a \to b) \leftrightarrow (\overline{b} \to \overline{a})$$

Có nghĩa là, để chứng minh mệnh đề $a \to b$ là đúng, ta có thể chứng minh mệnh đề $\overline{b} \to \overline{a}$.

Quy trình chứng minh. Để chứng minh một khẳng định theo phương pháp phản chứng, ta cần thực thi các bước sau:

- *Bước 1*. Giả sử hệ quả đã cho là sai.
- *Bước 2*. Chứng minh quy tắc suy luận từ hệ quả sai sẽ dẫn đến một giả thiết trái ngược với giả thiết đã cho ban đầu.
- *Bước 3*. Khẳng định đúng.

Ví dụ 1. Cho 7 đoạn thẳng có độ dài lớn hơn 10 và nhỏ hơn 100. Chứng minh rằng ta luôn luôn tìm được 3 đoạn để có thể ghép lại thành một tam giác.

Giải. Điều kiện cần và đủ để 3 đoạn là cạnh của một tam giác là tổng của hai cạnh phải lớn hơn một cạnh. Ta sắp các đoạn thẳng theo thứ tự tăng dần của độ dài $a_1, a_2, \ldots, a_7$ và chứng minh rằng dãy đã xếp luôn tìm được 3 đoạn mà tổng của hai đoạn đầu lớn hơn đoạn cuối.

Bước 1. Giả sử không tìm được ba đoạn nào mà tổng của hai đoạn nhỏ hơn một đoạn.

Bước 2. Theo tính chất của bất đẳng thức tam giác:

$$a_1 + a_2 \leq a_3 \Rightarrow a_3 \geq 20 \ (vì\ a_1, a_2 \geq 10)$$

$$a_2 + a_3 \leq a_4 \Rightarrow a_4 \geq 30 \ (vì\ a_2 \geq 10, a_3 \geq 20)$$

$$a_3 + a_4 \leq a_5 \Rightarrow a_5 \geq 50 \ (vì\ a_3 \geq 20, a_4 \geq 30)$$

$$a_4 + a_5 \leq a_6 \Rightarrow a_6 \geq 80 \ (vì\ a_4 \geq 30, a_5 \geq 50)$$

$$a_5 + a_6 \leq a_7 \Rightarrow a_7 \geq 130 \ (vì\ a_5 \geq 50, a_6 \geq 80)$$

⇒ Mâu thuẫn (vì tất cả các cạnh đều có độ dài nhỏ hơn 100).

Bước 3. Vậy, ta luôn tìm được 3 đoạn thẳng thỏa mãn yêu cầu.

Ví dụ 2. Các đỉnh của một thập giác đều được đánh số bởi các số nguyên 0, 1,.., 9 một cách tuỳ ý. Chứng minh rằng luôn tìm được ba đỉnh liên tiếp có tổng các số là lớn hơn 13.

Giải. Gọi x_1, x_2,.., x_{10} là các số gán cho các đỉnh của thập giác đều.

Bước 1. Giả sử ngược lại ta không tìm được 3 đỉnh liên tiếp nào thoả mãn khẳng định trên.

Bước 2. Theo giả thiết ta có:

$$x_1 + x_2 + x_3 \leq 13$$
$$x_2 + x_3 + x_4 \leq 13$$
$$x_3 + x_4 + x_5 \leq 13$$
$$x4 + x5 + x6 \leq 13$$
$$x_5 + x_6 + x_7 \leq 13$$
$$x_6 + x_7 + x_8 \leq 13$$
$$x_7 + x_8 + x_9 \leq 13$$
$$x_8 + x_9 + x_{10} \leq 13$$
$$x_9 + x_{10} + x_1 \leq 13$$
$$x_{10} + x_1 + x_2 \leq 13$$

Cộng các bất đẳng thức trên theo vế:

$$VT = 3\,(x_1 + x_2 + \ldots + x_{10}) = 3\,(0 + 1 + 2 + \ldots + 9) = 135$$

$VP = 130$

Bất đẳng thức $135 \leq 130$ sai, suy ra mâu thuẩn.

Bước 3. Khẳng định được chứng minh.

3.2. Nguyên lý Dirichlet

3.2.1. Phát biểu nguyên lý

Giả sử có một đàn chim bồ câu bay vào chuồng. Nếu số chim nhiều hơn số ngăn chuồng thì ít nhất trong một ngăn có nhiều hơn một con chim. Nguyên lý này dĩ nhiên là có thể áp dụng cho các đối tượng không phải là chim bồ câu và chuồng chim.

Nguyên lý. Nếu có k+1 (hoặc nhiều hơn) đồ vật được đặt vào trong k hộp thì tồn tại một hộp có ít nhất hai đồ vật.

Chứng minh. Chúng ta sử dụng phương pháp phản chứng. Giả sử rằng mỗi hộp chứa tối đa là 1 đồ vật. Ta có k hộp, nghĩa là có k đồ vật. Điều này trái với giả thiết có k+1 đồ vật.

<u>Các ví dụ</u>

Ví dụ 1. Trong bất kỳ một nhóm 367 người thế nào cũng có ít nhất hai người có ngày sinh nhật giống nhau bởi vì chỉ có tất cả 366 ngày sinh nhật khác nhau.

Ví dụ 2. Trong kỳ thi học sinh giỏi, điểm bài thi được đánh giá bởi một số nguyên trong khoảng từ 0 đến 100. Hỏi rằng ít nhất có bao nhiêu học sinh dự thi để cho chắc chắn tìm được hai học sinh có kết quả thi như nhau? Theo nguyên lý Dirichlet, số học sinh cần tìm là 102, vì ta có 101 kết quả điểm thi khác nhau.

3.2.2. Nguyên lý Dirichlet tổng quát

Nguyên lý. Nếu có N đồ vật được đặt vào trong k hộp thì sẽ tồn tại một hộp chứa ít nhất $\left\lceil \frac{N}{K} \right\rceil$ đồ vật. Trong đó, hàm $\lceil\ \ \rceil$ dùng để lấy trần của một số (nghĩa là số nguyên cận trên đúng của nó).

Cần lưu ý rằng, trong các phép toán làm tròn số, ta có

$[\ \ ]$ - Phép lấy phần nguyên.

$\lceil\ \ \rceil$ - Phép làm tròn lên (hay phép lấy trần) hay hàm $ceil()$.

$\lfloor\ \ \rfloor$ - Phép làm tròn xuống (hay phép lấy sàn) hay hàm $floor()$.

Các ví dụ

Ví dụ 1. Trong 100 người, có ít nhất 9 người sinh cùng một tháng.

Xếp những người sinh cùng tháng vào một nhóm. Một năm có 12 tháng. Vậy theo nguyên lý Dirichlet, tồn tại một nhóm có ít nhất $\left\lceil \frac{100}{12} \right\rceil = \lceil 8.333 \rceil = 9$ người.

Ví dụ 2. Có năm loại học bổng khác nhau. Hỏi rằng phải có ít nhất bao nhiêu sinh viên để chắc chắn rằng có ít ra là 6 người cùng nhận học bổng như nhau.

Gọi N là số sinh viên, khi đó $\left\lceil \frac{N}{5} \right\rceil = 6$ khi và chỉ khi 5 < N div 5 ≤ 6 hay 25 < N ≤ 30. Các số thỏa mãn điều kiện này bao gồm 26, 27, 28, 29 và 30. Nhưng yêu cầu tìm số sinh viên ít nhất, do đó số N cần tìm là 26.

Tổng quát: Nếu cần tìm N nhỏ nhất, sao cho $\left\lceil \frac{N}{a} \right\rceil = k$ thì $N = (k-1)a + 1$.

Ví dụ 3. Trong một hình vuông cạnh 1. Hãy chứng tỏ rằng nếu gieo vào đó 5 điểm thì có ít nhất hai điểm có khoảng cách nhỏ hơn $\frac{\sqrt{2}}{2}$.

Nếu ta chia hình vuông cạnh 1 thành 4 hình vuông con. Thì khoảng cách tối đa của hai điểm trong một hình vuông nhỏ không vượt quá độ dài của đường chéo là $\frac{\sqrt{2}}{2}$. Nếu ta gieo vào 5 điểm trong hình vuông, thì sẽ có ít nhất $\left\lceil \frac{5}{4} \right\rceil = 2$ điểm nằm trong một hình vuông nhỏ. Suy ra, sẽ luôn tồn tại hai điểm có khoảng cách nhỏ hơn $\frac{\sqrt{2}}{2}$.

Ví dụ 4. Cho hình thang cân, có đáy lớn là 2, đáy bé là 1, chiều cao là 1. Người ta gieo vào đó 13 điểm. Hãy chứng minh rằng, luôn tồn tại ít nhất 3 điểm mà diện tích của hình tạo bởi ba điểm đó nhỏ hơn ¼.

Dễ dàng chứng tỏ được rằng mỗi phần được chia như trên có diện tích ¼. Khi gieo vào 13 điểm, sẽ có ít nhất $\left\lceil \frac{13}{6} \right\rceil = 3$ điểm nằm trong 1 phần. Suy ra 3 điểm nằm trong một vùng diện tích ¼ thì diện tích tạo bởi chúng nhỏ hơn ¼.

CHƯƠNG 4. BÀI TOÁN LIỆT KÊ

4.1. Giới thiệu bài toán

Bài toán đưa ra danh sách tất cả các cấu hình tổ hợp có thể có được gọi là bài toán liệt kê tổ hợp. Khác với bài toán đếm là tìm kiếm một công thức cho lời giải, bài toán liệt kê lại cần xác định một thuật toán để theo đó có thể xây dựng được lần lượt tất cả các cấu hình cần quan tâm. Một thuật toán liệt kê phải đảm bảo hai nguyên tắc:

- Không được lặp lại một cấu hình
- Không được bỏ sót một cấu hình

Ví dụ 1. Cho tập hợp các số a_1, a_2,.., a_n và số M. Hãy tìm tất cả các tập con k phần tử của dãy số {a_n} sao cho tổng số các phần tử trong tập con đó đúng bằng M.

Giải. Như chúng ta đã biết, số các tập con k phần tử của tập gồm n phần tử là C_n^k. Như vậy chúng ta cần phải duyệt trong số C_n^k tập k phần tử để lấy ra những tập có tổng các phần tử đúng bằng M. Vì không thể xác định được có bao nhiêu tập k phần tử từ tập n phần tử có tổng các phần tử đúng bằng M nên chúng ta chỉ còn cách liệt kê các cấu hình thoả mãn điều kiện đã cho.

Ví dụ 2. Một thương nhân đi bán hàng tại tám thành phố. Chị ta có thể bắt đầu hành trình của mình tại một thành phố nào đó nhưng phải qua 7 thành phố kia theo bất kỳ thứ tự nào mà chị ta muốn. Hãy chỉ ra lộ trình ngắn nhất mà chị ta có thể đi.

Giải. Vì thành phố xuất phát đã được xác định. Do vậy thương nhân có thể chọn tuỳ ý 7 thành phố còn lại để hành trình. Như vậy, tất cả số hành trình của thương nhân có thể đi qua là 7! = 5040 cách. Tuy nhiên trong 5040 cách chúng ta phải duyệt toàn bộ để chỉ ra một hành trình là ngẵn nhất.

Có thể nói phương pháp liệt kê là biện pháp cuối cùng nhưng cũng là biện pháp phổ dụng nhất để giải quyết các bài toán tổ hợp. Khó khăn chính của phương pháp này là sự bùng nổ tổ hợp. Để xây dựng chừng 1 tỷ cấu hình, giả sử cần 1 giây để liệt kê một cấu hình thì chúng ta cũng cần 31 năm mới giải quyết xong. Tuy nhiên với sự phát triển nhanh chóng của máy tính, bằng phương pháp liệt kê, nhiều bài toán khó của lý

thuyết tổ hợp đã được giải quyết và góp phần thúc đẩy sự phát triển của nhiều ngành toán học.

4.2. Thuật toán quay lui

Để giải quyết những bài toán tổ hợp phức tạp, người ta thường dùng thuật toán quay lui (Back Track) sẽ được trình bày dưới đây.

Nội dung chính của thuật toán này là xây dựng dần các thành phần của cấu hình bằng cách thử tất cả các khả năng. Giả sử cần phải tìm một cấu hình của bài toán x = (x_1, x_2,.., x_n) mà i-1 thành phần x_1, x_2,.., x_{i-1} đã được xác định, bây giờ ta xác định thành phần thứ i của cấu hình bằng cách duyệt tất cả các khả năng có thể có và đánh số các khả năng từ 1..n_i. Với mỗi khả năng j, kiểm tra xem j có chấp nhận được hay không. Khi đó có thể xảy ra hai trường hợp:

- Nếu chấp nhận j thì xác định x_i theo j, nếu i=n thì ta được một cấu hình cần tìm, ngược lại xác định tiếp thành phần x_{i+1}.
- Nếu thử tất cả các khả năng mà không có khả năng nào được chấp nhận thì quay lại bước trước đó để xác định lại x_{i-1}.

Điểm quan trọng nhất của thuật toán là phải ghi nhớ lại mỗi bước đã đi qua, những khả năng nào đã được thử để tránh sự trùng lặp. Để nhớ lại những bước duyệt trước đó, chương trình cần phải được tổ chức theo cơ chế ngăn xếp (Last in first out). Vì vậy, thuật toán quay lui rất phù hợp với những phép gọi đệ qui. Thuật toán quay lui xác định thành phần thứ i có thể được mô tả bằng thủ tục Try(i) như sau:

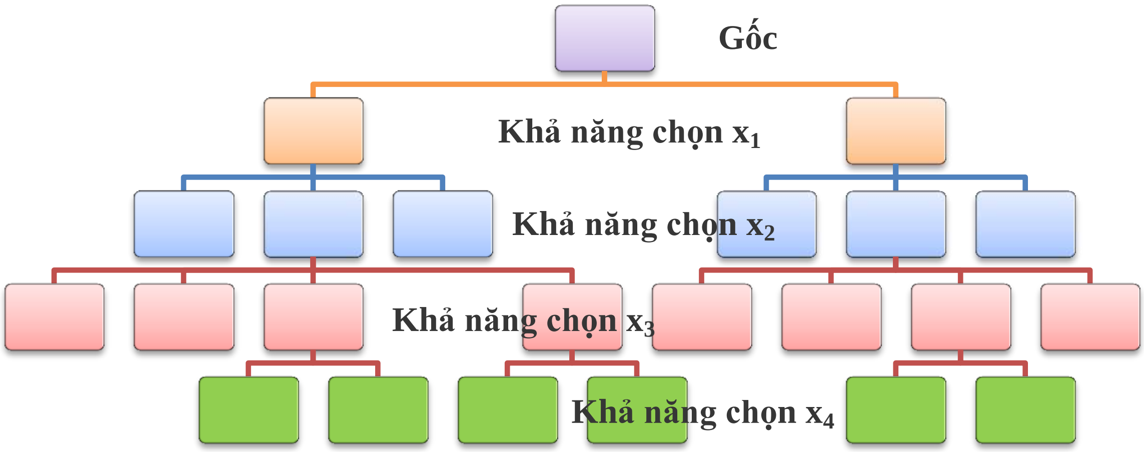
```
void Try( int i ) {
  int j;
  for ( j = 1; j < n_i; j ++) {
        if ( <Chấp nhận j >) {
            <Xác định xi theo j>
            if (i==n)
                    <Ghi nhận cấu hình>;
```

else Try(i+1);

}

}

Có thể mô tả quá trình tìm kiếm lời giải theo thuật toán quay lui bằng cây tìm kiếm lời giải sau:

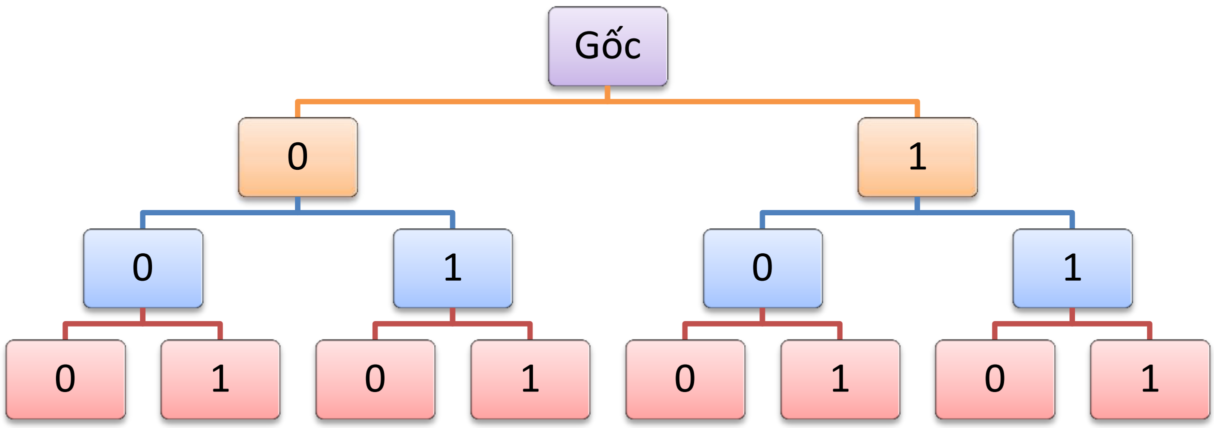


Hình 4.1 – Cây tìm kiếm thuật toán quay lùi

<u>Bài tập áp dụng</u>

Ví dụ 1. Liệt kê các xâu nhị phân độ dài n.

Giải. Biểu diễn các xâu nhị phân dưới dạng $b_1, b_2,..., b_n$, trong đó $b_i \in \{0, 1\}$. Thủ tục đệ qui Try(i) xác định b_i với các giá trị đề cử cho b_i là 0 và 1. Chẳng hạn với n =3, cây tìm kiếm lời giải được thể hiện như sau.

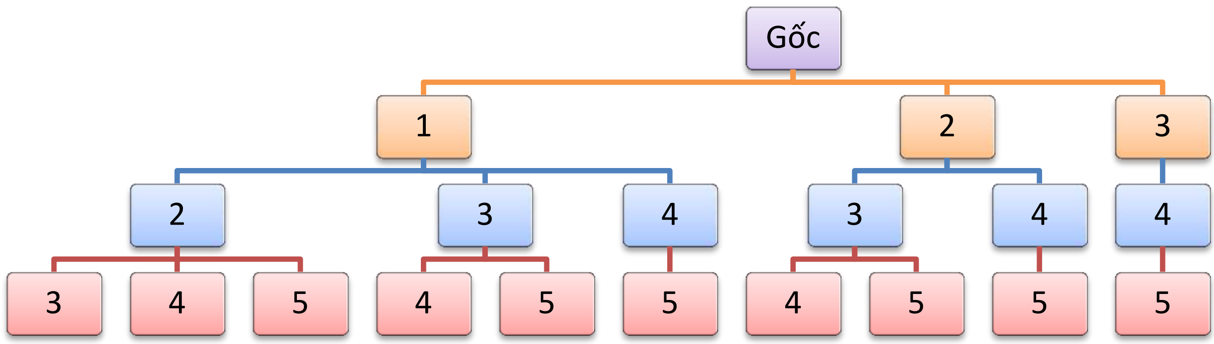


Hình 4.2 – Cây tìm kiếm xâu nhị phân độ dài 3

Ví dụ 2. Liệt kê các tập con k phần tử của tập n phần tử.

Giải. Biểu diễn tập con k phần tử dưới dạng c_1, c_2,.., c_k, trong đó 1< c_1<c_2..≤n. Từ đó suy ra các giá trị đề cử cho c_i là từ c_{i-1} + 1 cho đến n - k + i. Cần thêm vào c_0 = 0.

Cây tìm kiếm lời giải bài toán liệt kê tập con k phần tử của tập n phần tử với n=5, k=3 được thể hiện như dưới đây.



Hình 4.3 – Cây tìm kiếm các tập con 3 phần tử của tập 5 phần tử

Ví dụ 4. *Bài toán Xếp Hậu*. Liệt kê tất cả các cách xếp n quân hậu trên bàn cờ $n \times n$ sao cho chúng không ăn được nhau.

Giải. Bàn cờ có n hàng được đánh số từ 0 đến n-1, n cột được đánh số từ 0 đến n-1; Bàn cờ có n*2 -1 đường chéo xuôi được đánh số từ 0 đến 2*n -2, 2 *n -1 đường chéo ngược được đánh số từ 2*n -2. Ví dụ: với bàn cờ 8 x 8, chúng ta có 8 hàng được đánh số từ 0 đến 7, 8 cột được đánh số từ 0 đến 7, 15 đường chéo xuôi, 15 đường chéo ngược được đánh số từ 0..15.

Vì trên mỗi hàng chỉ xếp được đúng một quân hậu, nên chúng ta chỉ cần quan tâm đến quân hậu được xếp ở cột nào. Từ đó dẫn đến việc xác định bộ n thành phần x_1, x_2,.., x_n, trong đó x_i = j được hiểu là quân hậu tại dòng i xếp vào cột thứ j. Giá trị của i được nhận từ 0 đến n-1; giá trị của j cũng được nhận từ 0 đến n-1, nhưng thoả mãn điều kiện ô (i,j) chưa bị quân hậu khác chiếu đến theo cột, đường chéo xuôi, đường chéo ngược.

Việc kiểm soát theo hàng ngang là không cần thiết vì trên mỗi hàng chỉ xếp đúng một quân hậu. Việc kiểm soát theo cột được ghi nhận nhờ dãy biến logic a_j với qui ước a_j=1 nếu cột j còn trống, cột a_j=0 nếu cột j không còn trống. Để ghi nhận đường chéo xuôi và đường chéo ngược có chiếu tới ô (i,j) hay không, ta sử dụng phương trình i + j = const và i - j = const, đường chéo thứ nhất được ghi nhận bởi dãy biến b_j, đường chéo thứ 2 được ghi nhận bởi dãy biến c_j với qui ước nếu đường chéo nào còn trống thì giá trị tương ứng của nó là 1 ngược lại là 0. Như vậy, cột j được chấp nhận khi cả 3 biến a_j, b_{i+j}, c_{i+j} đều có giá trị 1. Các biến này phải được khởi đầu giá trị 1 trước đó, gán lại giá trị 0 khi xếp xong quân hậu thứ i và trả lại giá trị 1 khi đưa ra kết quả.

Dưới đây là số cách xếp hậu ứng với n.

n	4	7	8	9	10	11	12	13	14
H_n	2	40	92	352	724	2680	14200	73712	365596

CHƯƠNG 5. BÀI TOÁN TỐI ƯU

5.1. Phát biểu bài toán

"*Trong tất cả các cấu hình tổ hợp chấp nhận được của một bài toán tổ hợp, hãy tìm cấu hình tổ hợp có giá trị sử dụng tốt nhất (theo một mục đích nào đó)*".

Bài toán ta vừa nêu là bài toán tối ưu tổ hợp. Tối ưu tổ hợp cũng là một phân ngành của Tối ưu hóa và cũng là một phân ngành của Toán rời rạc. Bài toán tối ưu tổ hợp có thể phát biểu tổng quát như sau: Tìm cực tiểu (hay cực đại) của hàm: $f(x) \rightarrow \min$ (hoặc max), với $x \in D$; trong đó D là tập hữu hạn phần tử.

Hàm f(x) được gọi là hàm mục tiêu của bài toán, mỗi phần tử $x \in D$ được gọi là một phương án, còn tập D gọi là tập các phương án của bài toán. Phần tử $x \in D$ mà với phần tử này, hàm mục tiêu đạt giá trị tối ưu được gọi là phương án tối ưu.

5.2. Thuật toán nhánh và cận

5.2.1. Bài toán

Tìm $\min\{f(x) : x \in D\}$ với D là tập hữu hạn phần tử.

Giả sử D được mô tả như sau:

$$D = \{x = (x_1, x_2, \dots, x_n) \in A_1 \times A_2 \times \dots \times A_n : x \text{ thoả mãn tính chất } P\},$$

với $A_1 \times A_2 \times \dots \times A_n$ là các tập hữu hạn, còn P là tính chất cho trên tích Đềcác $A_1 \times A_2 \times \dots \times A_n$.

Thuật toán nhánh và cận có thể giải bài toán này nếu như có thể tìm được một hàm g thoả mãn bất đẳng thức sau:

$$g(a_1, a_2, \dots, a_k) \le \min\{f(x) : x \in D, x_i = a_i, i = 1,2,\dots,k\}, \text{ với } k = 1,2,\dots$$

5.2.2. Thuật toán nhánh và cận

void Try(k)

{

for $a_k \in A_k$

```
            if <chấp nhận a_k>
                {
                    x_k = a_k;
                    if (k==n) <Cập nhật kỷ lục>;
                    else Try(k+1);
                }
}

void Nhanh_can()
{
    f̄ = +∞;
    Try(1);
    if (f̄ < +∞)  <f̄ là giá trị tối ưu, x̄ là phương án tối ưu>;
    else <Bài toán không có phương án>;
}
```

5.2.3. Ví dụ

a. Bài toán cái túi

Phát biểu. Có n loại đồ vật, đồ vật thứ j có trọng lượng a_j và giá trị sử dụng là c_j $(j = 1,2, \dots, n)$. Cần chất các đồ vật này vào một cái túi có khả năng chứa trọng lượng tối đa là b sao cho tổng giá trị sử dụng của các đồ vật chất trong túi là lớn nhất.

Mô hình toán. Giả sử rằng $x = (x_1, x_2, \dots, x_n)$ là vector trạng thái của các đồ vật. Thành phần x_j có khả năng nhận một trong hai giá trị là 0 hoặc 1. Nếu x_j nhận giá trị 0, có nghĩa là vật thứ *j* sẽ không có trong túi, ngược lại x_j nhận giá trị 1, có nghĩa là vật thứ j sẽ có trong túi. Nếu x_j nhận một giá trị lớn hơn 1, có nghĩa là đồ vật thứ *j* sẽ chứa nhiều hơn 1 lần trong túi.

Gọi *f(x)* là giá trị sử dụng của các đồ vật. Khi đó, ta có

$$f(x) = \sum_{i=1}^{n} c_i x_i$$

Hàm này được gọi là hàm mục tiêu.

Trọng lượng của các đồ vật có trong túi là

$$\sum_{i=1}^{n} a_i x_i \leq b$$

Theo yêu cầu của bài toán, ta cần tìm bộ $x = (x_1, x_2, \dots, x_n)$ sao cho hàm $f(x) \rightarrow max$.

Do đó, mô hình toán của bài toán cái túi nhận được là

$$\begin{cases} f(x) = \sum_{i=1}^{n} c_i . x_i \rightarrow max, (x_1, x_2, \dots, x_n) \in D \\ \sum_{i=1}^{n} a_i . x_i \leq b, x_i \in [0, 1] \end{cases}$$

Có nhiều phương pháp để giải bài toán cái túi: phương pháp lặp đơn hình (Simplex method), giải thuật tham lam (greedy approximation algorithm)... Tuy nhiên, trong chương này, chúng ta chỉ nghiên cứu phương pháp nhánh cận.

Phương pháp giải. Ta cần tìm hàm $f^*(x)$ sao cho

$$\begin{cases} f^*(x) = \sum_{i=1}^{n} c_i . x_i \rightarrow max, (x_1, x_2, \dots, x_n) \in Z^+ \\ \sum_{i=1}^{n} a_i . x_i \leq b \end{cases} \quad (1)$$

Kí hiệu M là tập các phương án của bài toán (1).

Không giảm tổng quát ta giả thiết rằng các đồ vật được đánh số sao cho bất đẳng thức sau được thoả mãn:

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \dots \geq \frac{c_n}{a_n} \quad (2)$$

Để xây dựng hàm tính cận dưới, cùng với bài toán cái túi (1), ta xét bài toán cái túi biến liên tục sau:

$$\begin{cases} f^*(x) = \sum_{i=1}^{n} c_i . x_i \rightarrow max, (x_1, x_2, \dots, x_n) \in N \\ \sum_{i=1}^{n} a_i . x_i \leq b \end{cases} \quad (3)$$

Lưu ý. Tập Z^+ là tập số nguyên dương (không chứa số 0), tập N là tập số tự nhiên (chứa số 0).

Mệnh đề. *Phương án tối ưu của bài toán (3) là vectơ $\overline{x} = (\overline{x_1}, \overline{x_2}, \dots, \overline{x_n})$ với các thành phần được xác định bởi công thức:*

$$\overline{x_1} = \frac{b}{a_1}, \overline{x_2} = \overline{x_3} = \dots = \overline{x_n} = 0$$

và giá trị tối ưu là $g^* = \frac{c_1 . b_1}{a_1}$.

Giả sử ta có phương án bộ phận cấp k là $\bar{x} = (\overline{x_1}, \overline{x_2}, \dots, \overline{x_k})$. Khi đó giá trị sử dụng của các đồ vật đang có trong túi là:

$$\partial_k = \sum_{i=1}^{k} c_i . \overline{x_i}$$

và trọng lượng còn lại của cái túi là:

$$w_k = b - \sum_{i=1}^{k} a_i . \overline{x_i}$$

Khi đó, công thức (3) có thể viết lại như sau:

$$f^*(x) = \max\left\{\sum_{i=1}^{n} c_i \overline{x}_i : \sum_{i=1}^{n} a_i \overline{x}_i \leq b, \overline{x}_i \in [0, 1]\right\}$$

$$= \max\left\{\sum_{i=1}^{k} c_i \overline{x}_i + \sum_{i=k+1}^{n} c_i \overline{x}_i : \sum_{i=1}^{k} a_i \overline{x}_i + \sum_{i=k+1}^{n} a_i \overline{x}_i \leq b, \overline{x}_i \in [0, 1]\right\}$$

$$= \max\left\{\partial_k + \sum_{i=k+1}^{n} c_i\overline{x}_i : \sum_{i=k+1}^{n} a_i\overline{x}_i \le w_k, \overline{x}_i \in [0,1]\right\}$$

Vì giá trị $\overline{x}$ là phương án tối ưu bộ phận cấp k, nên giá trị ∂_k là cố định, do đó

$$f^*(x) = \partial_k + \max\left\{\sum_{i=k+1}^{n} c_i\overline{x}_i : \sum_{i=k+1}^{n} a_i\overline{x}_i \le w_k, \overline{x}_i \in [0,1]\right\}$$

Khi chuyển sang bước lặp tiếp theo, ta xét trường hợp $x_{k+1} \neq 0, x_j = 0, \forall j > k+1$, khi đó:

$$f^*(x) = \partial_k + \max\{c_{k+1}\overline{x}_{k+1} : a_{k+1}\overline{x}_k \le w_k\}$$

$$= \partial_k + \frac{c_{k+1}}{a_{k+1}} w_k$$

Đặt

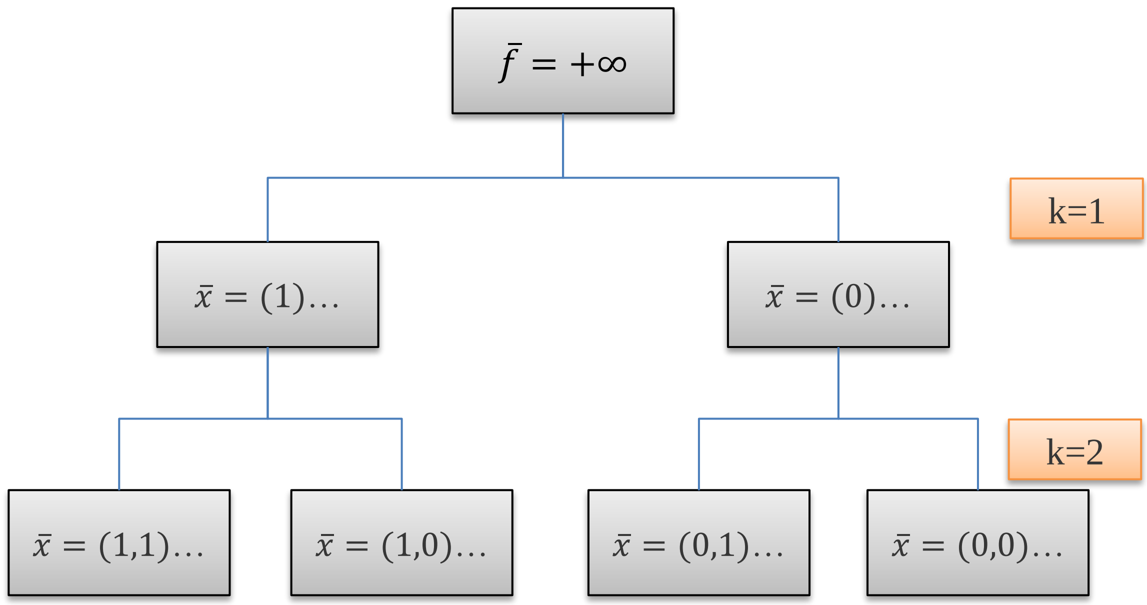
$$g_k = \partial_k + \frac{c_{k+1}}{a_{k+1}} w_k$$

và công thức này cho phép ta tính cận trên cho phương án bộ phận cấp k.

Mô tả thuật toán

Bước 1. Đặt $\bar{f} = +\infty, \bar{x} = (\overline{x_1}, \dots, \overline{x_k}, \dots, \overline{x_n})$. Khởi tạo giá trị k=1.

Bước 2. Thực hiện thuật toán rẽ nhánh theo phương án tối ưu bộ phận từ gốc $\bar{f}$. Tương ứng với gốc này, ta rẽ nhánh theo các phần tử $\overline{x_1}, \overline{x_2}, \dots, \overline{x_n}$.



Hình 5.1 – Cây tìm kiếm thuật toán nhánh cận cho bài toán cái túi

Bước 3. Tại các nút nhánh, ta thực hiện tính toán các giá trị

$$\partial_k = \sum_{i=1}^{k} c_i.\bar{x}_i$$

$$w_k = b - \sum_{i=1}^{k} a_i.\bar{x}_i$$

$$g_k = \partial_k + \frac{c_{k+1}}{a_{k+1}} w_k$$

tương ứng với các phương án bộ phận $\bar{x}$.

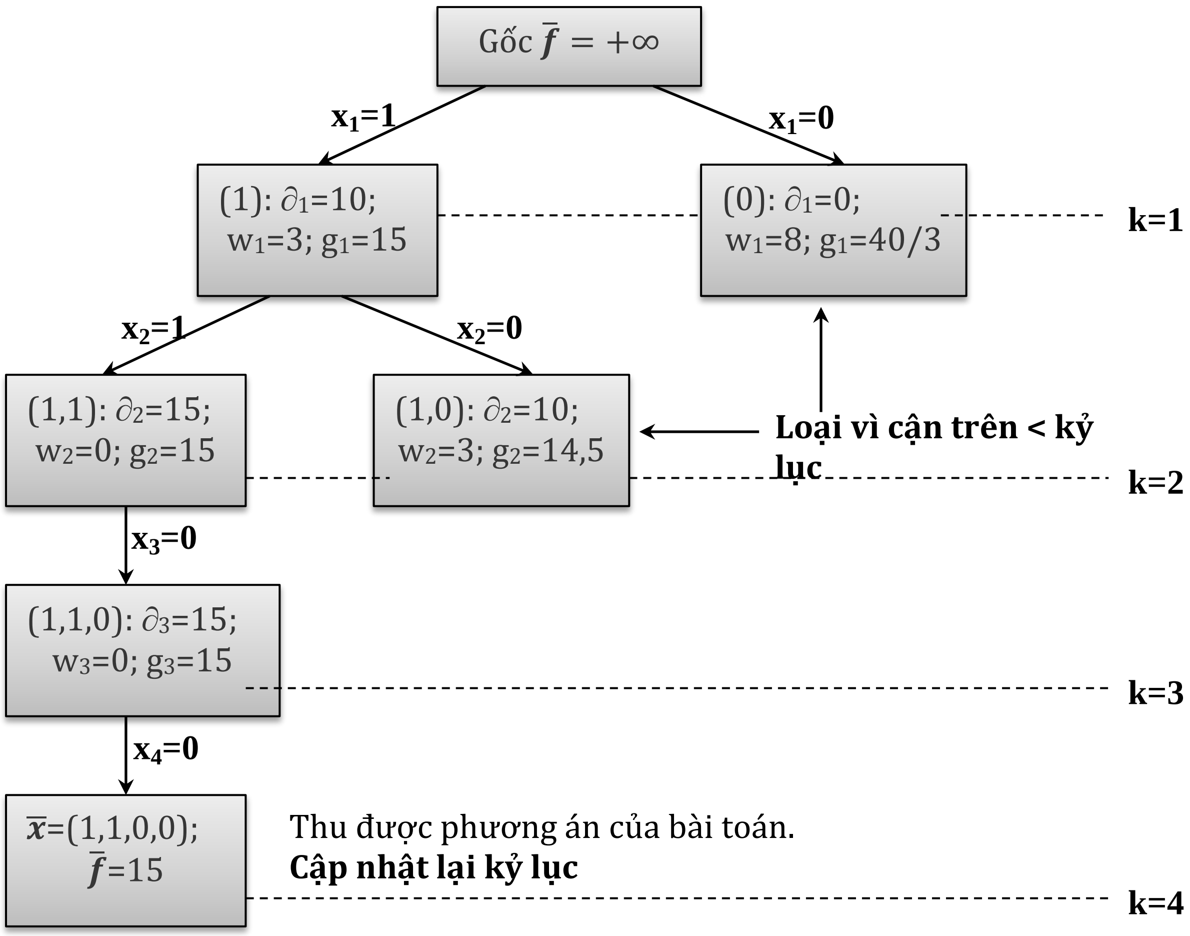
Bước 4. Tăng giá trị k và quay lại bước 2. Thuật toán sẽ dừng nếu giá trị k=n. Trong quá trình tính toán, nếu tại một nút nhánh nào đó, giá trị $w_k < 0$ hoặc $g_k < \partial_{max}$ thì ta sẽ không tiến hành rẽ nhánh theo nhánh này. Trong đó, ∂_{max} gọi là kỉ lục. Nó là giá trị lớn nhất trong các giá trị ∂_k mà ta đã tính toán trước đó. Giá trị kỉ lục này phải được cập nhật thường xuyên sau mỗi phương án được tìm thấy. Nếu tại một nút nào

đó, việc rẽ nhánh không được thực thi, ta không cần cập nhật lại kỉ lục trong bước lặp này.

Ví dụ. Giải bài toán cái túi sau theo thuật toán nhánh và cận

$$\begin{cases} f(x) = 10.x_1 + 5.x_2 + 3.x_3 + 6.x_4 \to max \\ \quad 5.x_1 + 3.x_2 + 2.x_3 + 4.x_4 \le 8 \end{cases}$$

Giải. Quá trình giải bài toán được mô tả trong cây tìm kiếm trong hình sau. Thông tin về một phương án bộ phận trên cây được ghi trong các ô trên hình vẽ.



Hình 5.2 – Tiến trình thực thi giải thuật nhánh cận cho bài toán cái túi

Kết thúc thuật toán, ta thu được phương án tối ưu là x^* = (1, 1, 0, 0) và giá trị tối ưu là f^* = 15.

b. Bài toán người du lịch

Phát biểu. Giả sử ta có n thành phố được đánh chỉ số là 1, 2,...n. Người đi du lịch xuất phát tại một thành phố bất kì cần đi qua tất cả các thành phố còn lại, mỗi thành phố đi qua đúng một lần và quay lại thành phố ban đầu. Hãy tìm một đường đi tối ưu nhất.

Thông thường nguyên lý tối ưu trong trường hợp này là tối ưu về tổng chi phí bỏ ra, cũng chính là tổng trọng số của các đường đi từ $i \to j$. Trọng số này có thể hiểu là khoảng cách về không gian hay chi phí đường đi... Vì vậy, với n thành phố, thì bài toán du lịch sẽ dẫn đến một ma trận trọng số $\left(c_{i,j}\right)_{n\times n}$ – trong đó mỗi phần tử c_{ij} là chi phí bỏ ra để đi từ thành phố $i \to j$. Một cách tự nhiên, các phần tử trên đường chéo của ma trận chi phí sẽ không xác định.

Bài toán người du lịch có một ứng dụng rất lớn trong thực tiễn, rất nhiều bài toán đếm trong lí thuyết tổ hợp đều được đưa về dạng bài toán này.

Thuật toán nhánh cận để giải bài toán người du lịch. Nếu sử dụng phương pháp liệt kê, ta cần khảo sát đến n! cấu hình tổ hợp. Tuy nhiên, trong lộ trình đi, người du lịch xuất phát tại một thành phố bất kì và quay trở lại thành phố ban đầu nên nếu ta cố định một thành phố ban đầu là i, thì ta chỉ phải xét đến $(n-1)!$ cấu hình tổ hợp.

Mô hình toán của bài toán người du lịch

Gọi $f(1,\dots,n)$ là hàm mục tiêu tương ứng với độ dài của lộ trình. Khi đó bài toán sẽ biểu diễn dưới dạng mô hình toán sau:

$$f(1,\dots,n) = \sum_{i=2}^{n} c_{i-1,i} \to min$$

Phương pháp giải

Gọi

$$c_{\min} \quad = \min\left\{c_{ij} \middle| i,j = 1,\dots n, i \neq j\right\}$$

là cận dưới chi phí.

Giả sử ta đã tìm ra được k thành phố và khi đó, hành trình bộ phận qua k thành phố này là

$$(u_1,u_2) \to (u_2,u_3) \to \cdots \to (u_{k-1},u_k)$$

Trong đó, $u_i, i = 1,..,k$ là k thành phố trong n thành phố $1,\dots,n$.

Chi phí cho hành trình bộ phận này là

$$\partial(1, \dots, k) = f(1, \dots, k) = \sum_{i=2}^{k} c_{i-1,i}$$

Để xây dựng một hành trình hoàn chỉnh, ta cần đi qua n-k thành phố còn lại và sau đó quay lại thành phố ban đầu. Nghĩa là cần đi qua thêm n-k+1 thành phố. Do đó, cận dưới của phương án bộ phận này là

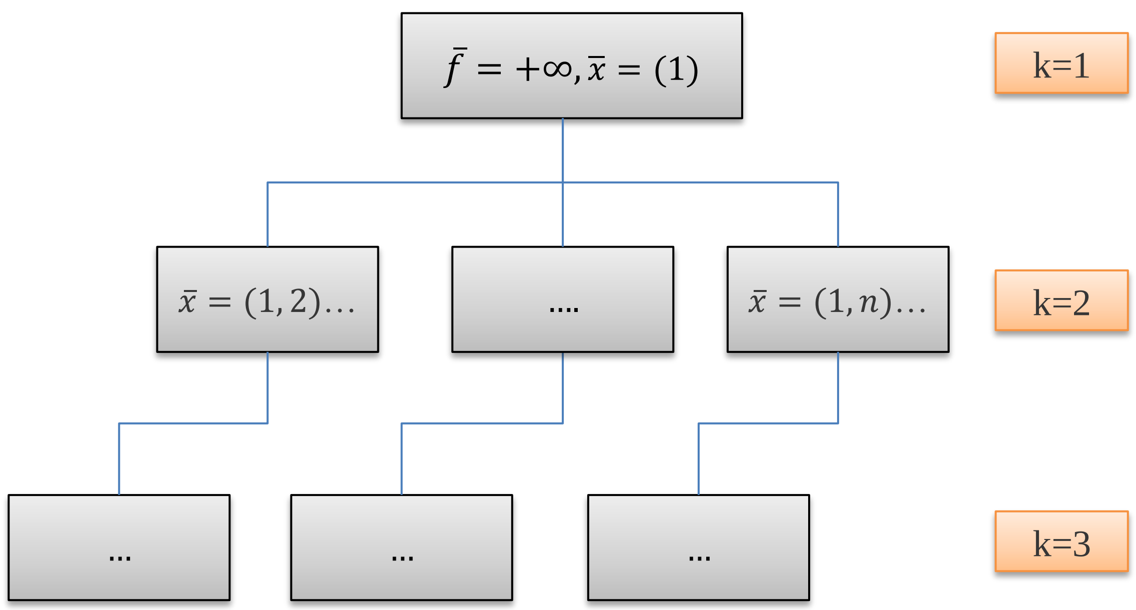
$$g(u_1, \dots, u_k) = f(1, \dots, k) + (n-k+1).c_{\min}$$

Để tránh rườm rà, trong thuật toán, ta sẽ sử dụng g và ∂ để thay thế cho $g(u_1, \dots, u_k)$ và $\partial(u_1, \dots, u_k)$.

Thuật toán

Bước 1. Cố định đỉnh 1, tính $c_{min} = min\{c_{ij} | i, j = 1..n, i \neq j\}$. Đặt $\bar{f} = +\infty, \bar{x} = (1)$. Khởi tạo giá trị k=1.

Bước 2. Thực hiện thuật toán rẽ nhánh theo phương án tối ưu bộ phận từ gốc $\bar{f}$. Tương ứng với gốc này, ta rẽ nhánh theo các phần tử $2, 3, \dots, n$.



Hình 5.3 – Cây tìm kiếm thuật toán nhánh cận cho bài toán người du lịch

Bước 3. Tại các nút nhánh, ta thực hiện tính toán các giá trị

$$\partial_k = \sum_{i=2}^{k} c_{i-1,i}$$

$$g_k = \partial_k + (n - k + 1)c_{\min}$$

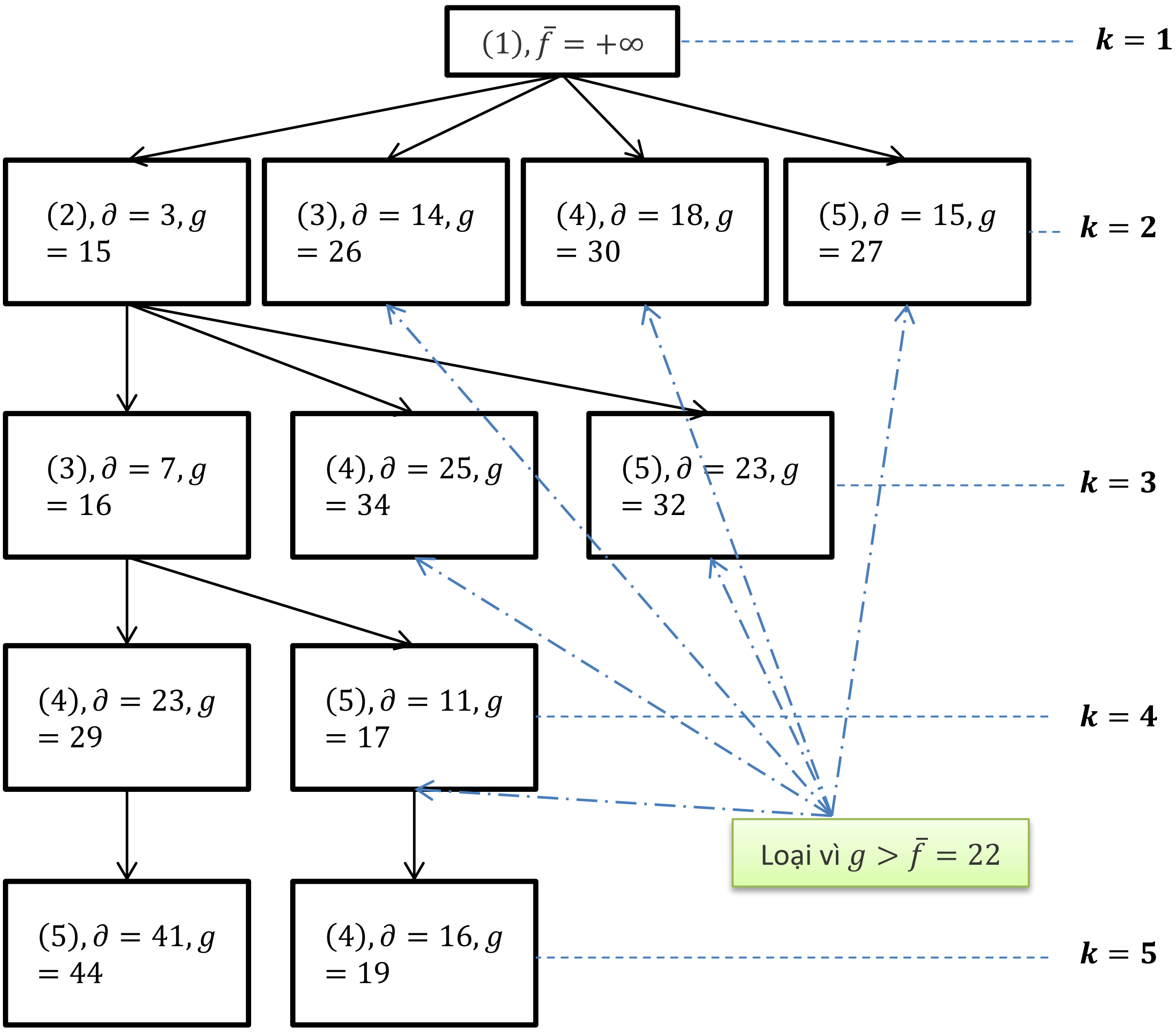
tương ứng với các phương án bộ phận $\bar{x}$.

*Bước 4***.** Tăng giá trị k và quay lại bước 2. Thuật toán sẽ dừng nếu giá trị k=n. Trong quá trình tính toán, nếu tại một nút nhánh nào đó, giá trị $g_k > \partial_{max}$ thì ta sẽ không tiến hành rẽ nhánh theo nhánh này. Trong đó, ∂_{max} gọi là kỉ lục. Nó là giá trị lớn nhất trong các giá trị ∂_k mà ta đã tính toán trước đó.

<u>Ví dụ</u>: Giải bài toán người du lịch với ma trận chi phí cho sau đây:

$$\begin{pmatrix} 0 & 3 & 14 & 18 & 15 \\ 3 & 0 & 4 & 22 & 20 \\ 17 & 9 & 0 & 16 & 4 \\ 6 & 2 & 7 & 0 & 12 \\ 9 & 15 & 11 & 5 & 0 \end{pmatrix}$$

Trong phương án (1, 2, 3, 4, 5, 1), ta có $\partial = 44$. Với phương án (1, 2, 3, 5, 4, 1), ta có $\partial = 22$. Do đó, ta chọn phương án tối ưu cục bộ là 22 – cũng là kỉ lục hiện tại. Các phương án còn lại bị loại như hình vẽ.



Hình 5.4 – Tiến trình thực thi giải thuật nhánh cận cho bài toán người du lịch

Kĩ thuật giảm nhánh cho thuật toán nhánh cận để giải bài toán người du lịch.

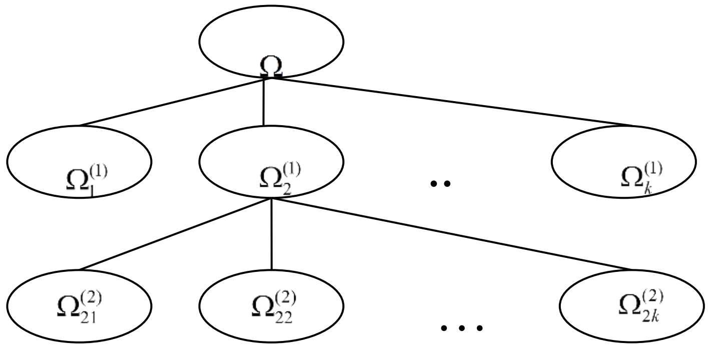
Giả sử chúng ta cần tìm giá trị tối thiểu của hàm mục tiêu

$$Z = f(x) \to \min_{x \in \Omega}$$

ở đó Ω – là tập hữu hạn các phương án X.

Để sử dụng phương pháp nhánh cận, ta cần thực hiện các bước chính sau đây

a) **Bước tạo nhánh**. Giải sử tồn tại một quy tắc để phân hoạch tập **Ω** thành các tập con: $\Omega_1^{(1)},\ldots,\Omega_k^{(1)}$ sao cho $\bigcup_{i=1}^{\pi}\Omega_i^{(1)} = \Omega$. Ta tiếp tục quá trình phân hoạch các tập con cho đến khi thu được tập con không thể phân hoạch được (gọi là đỉnh treo).



Hình 5.5 – Sơ đồ phân nhánh trong giải thuật thu gọn

b) **Tính cận dưới của hàm mục tiêu**. Với mỗi tập con Ω_k, ta nhận được kết quả phân nhánh để tính toán cận dưới của hàm mục tiêu trên tập con này. Nghĩa là $\xi(\Omega_k)$, sao cho

$$f(x) \geq \xi(\Omega_k) \forall x \in \Omega_k .$$

c) **Tính lại đánh giá**. Nếu hai tập Ω_1 và Ω_2 thỏa điều kiện $\Omega_1 \subset \Omega_2$, thì rõ ràng, hàm mục tiêu sẽ thỏa bất đẳng thức sau:

$$\min_{x \in \Omega_1} f(x) \geq \min_{x \in \Omega 2} f(x)$$

Chúng ta có thể coi rằng, với mỗi phân hoạch tập Ω' thành các tập con $\Omega'_1, \ldots, \Omega'_k$ thì phương pháp tính cận dưới cần thỏa mãn $\xi(\Omega'_i) \geq \xi(\Omega')$ nghĩa là, việc phân nhánh sẽ làm cho cận dưới luôn tăng.

d) **Xây dựng phương án**. Nếu trong một mắc xích nào đó của đồ thị phân nhánh, ta tìm được một đỉnh mà không thể tiếp tục phân nhánh, thì cần có một mắc xích để xây dựng nên phương án tương ứng cho nó.

Phương pháp cụ thể để xây dựng sẽ tùy thuộc vào các đặc trưng của mỗi bài toán.

e) **Dấu hiệu nhận biết tính tối ưu**. Đối với mỗi bài toán cụ thể, cần áp dụng nguyên lí sau: nếu tại một bước phân nhánh, ta tìm được một phương án tối ưu $\bar{X}$ và

$$f(\overline{X}) = \xi(\Omega_i) = \min \xi(\Omega_k),$$

ở đó, cực tiểu được chọn tương ứng với đỉnh treo của đồ thị phân nhánh, nên phương án này là phương án tối ưu. Và không thể có một cận dưới nhỏ hơn cận dưới của phương án tối ưu này.

f) **Phương pháp giảm thiểu việc phân nhánh**. Về cơ bản, chúng ta cần tìm một phương án nào đó trong tập Ω. Phương án này gọi là kỉ lục. Và chúng ta sẽ không khảo sát những đỉnh treo mà cận dưới của nó vượt kỉ lục.

Thuật toán nhánh cận để giải bài toán người du lịch.

Thuật toán này được giới thiệu đầu tiên bởi nhà toán học Lit năm 1963. Giả sử ma trận chi phí là $C = [c_{ij}]_{n\times n}$. Kí hiệu ∞ nghĩa là không tồn tại đường đi từ $i \to j$. Thuật toán sẽ thực hiện các bước sau đây:

Bước 1. Tìm trên mỗi dòng ma trận C phần tử nhỏ nhất và trừ tất cả các phần tử trên các dòng này cho phần tử nhỏ nhất tương ứng. Nếu ta nhận được một ma trận mà trên một cột bất kì không chứa phần tử 0 thì ta tiếp tục tìm phần tử nhỏ nhất trên mỗi cột và trừ tất cả các phần tử trên cột đó cho phần tử nhỏ nhất này. Kết quả, ta nhận được một ma trận $\boldsymbol{C'}$ mà mỗi dòng, mỗi cột của nó đều có ít nhất một phần tử 0.

Bước 2. Tính tổng của tất cả các phần tử đã trừ ở bước 1. Rõ ràng rằng, tổng này là cận dưới của tập Ω, mà ta sẽ chọn làm gốc để phân nhánh.

Bước 3. Chọn cung (x_k, x_l) sao cho

$$\theta(x_k, x_l) = \max_{(ij)} \gamma(x_k, x_l) \qquad , \tag{1}$$

ở đó $\gamma(x_k, x_l)$ – là tổng phần tử nhỏ nhất ở dòng i (trừ c'_{ij}) và phần tử nhỏ nhất ở cột j (trừ c'_{ij}) mà $c'_{ij} = 0$.

Xét tính chất $\overline{P_{ij}}$: Lộ trình không chứa cung (x_i, x_j) và Lộ trình chứa cung (x_i, x_j). Đối với lộ trình không chứa cung (x_i, x_j) ta sẽ bỏ đi dòng i và cột j. Còn đối với lộ trình chứa cung (x_i, x_j) ta sẽ thay phần tử $c'_{ij} = \infty$.

Bước 4. Tính $\theta(x_k, x_l)$ theo công thức (1) và cộng thêm nó vào cận của đỉnh tương ứng của cây bên phải. Tổng này sẽ là cận cho đỉnh mới được xác định theo tính chất $\overline{P_{kl}}$.

Bước 5. Tương tự, như ở bước 4, ta nối với đỉnh ở cây bên phải xác định theo tính chất P_{kl}: Lộ trình sử dụng cung (x_k, x_l). Xóa dòng k và cột l của ma trận và thay bằng phần tử ∞ cho giá trị tương ứng với cung này.

Bước 6. Thực hiện như bước 1 đối với ma trận mới thu được ở bước 5.

Bước 7. Thực hiện như ở bước 2 đối với ma trận nhận được ở bước 6. Bổ sung tổng nhận được thêm cho cận.

Bước 8. Nếu trong bước 5 ta nhận được ma trận bậc 1, thì tiến trình kết thúc. Ngược lại, chuyển sang bước 9.

Bước 9. Trong số các đỉnh treo của cây bên phải đã xây dựng, ta chọn đỉnh có cận nhỏ nhất (nếu có nhiều đỉnh có cùng cận, ta chọn một đỉnh bất kì).

Bước 10. Nếu chọn được một đỉnh ở bước 9 thỏa tính chất P_{ij}, thì ta chuyển sang bước 3, ngược lại chuyển sang bước 11.

Bước 11. Giá trị của phần tử ij của ma trận tương ứng với đỉnh treo được chọn ở bước 10 chuyển thành ∞. Tại dòng thứ i cũng như cột j, ta tìm phần tử bé nhất và trừ tất cả các phần tử của dòng hoặc cột này cho nó. Chuyển sang bước 3.

Ví dụ.

Đầu tiên, ta tìm trên mỗi dòng của ma trận phần tử nhỏ nhất.

i j	**1**	**2**	**3**	**4**	**5**	**min**
1	M	90	80	**40**	100	40
2	**60**	M	40	50	70	40
3	50	30	M	60	**20**	20
4	10	70	**20**	M	50	10
5	20	**40**	50	20	M	20

Sau đó, trừ tất cả các phần tử trên mỗi dòng cho phần tử nhỏ nhất của dòng đó. Sau bước này, ta nhận được ma trận mà mỗi dòng đều có phần tử 0.

i j	**1**	**2**	**3**	**4**	**5**
1	M	50	40	0	60
2	20	M	0	10	30
3	30	10	M	40	0
4	0	60	10	M	40

5	0	20	30	0	M

Thực hiện quá trình trên đối với cột.

i j	**1**	**2**	**3**	**4**	**5**
1	M	50	40	0	60
2	20	M	0	10	30
3	30	10	M	40	0
4	0	60	10	M	40
5	0	20	30	0	M
d_j	0	10	0	0	0

i j	**1**	**2**	**3**	**4**	**5**
1	M	40	40	0	60
2	20	M	0	10	30
3	30	0	M	40	0
4	0	50	10	M	40
5	0	10	30	0	M

Sau khi thực hiện quá trình này, ta nhận được ma trận có các dòng và cột đều chứa phần tử 0.

Cận dưới $\sum$=140. Tương ứng với mỗi đỉnh khi phân nhánh, ta có hai tập con là tập chứa cạnh đó và tập không chứa cạnh đó. Tại mỗi ô có giá trị 0 của ma trận, ta tính một hằng số là tổng của phần tử nhỏ nhất theo dòng và nhỏ nhất theo cột tương ứng với nó. Giá trị hằng này được ghi trong dấu ngoặc đơn.

i j	**1**	**2**	**3**	**4**	**5**	**d_i**
1	M	40	40	0(40)	60	40
2	20	M	0(20)	10	30	10

3	30	0(10)	M	40	0(30)	0
4	0(10)	50	10	M	40	10
5	0(0)	10	30	0(0)	M	0
d_j	0	10	10	0	30	0

d(1,4) = 40 + 0 = 40; d(2,3) = 10 + 10 = 20; d(3,2) = 0 + 10 = 10; d(3,5) = 0 + 30 = 30; d(4,1) = 10 + 0 = 10; d(5,1) = 0 + 0 = 0; d(5,4) = 0 + 0 = 0; Giá trị hằng lớn nhất là 40 tương ứng với (1,4). Nghĩa là tập đỉnh phân thành hai tập con: tập chứa cạnh (1, 4) và tập không chứa cạnh (1, 4). Cận dưới của tập con chứa cạnh (1, 4) là 140 + 40 = 180. Đối với tập không chứa cạnh (1, 4), để loại cạnh (1, 4), ta cần thay phần tử 0 tại ô này bằng ∞.

i j	**1**	**2**	**3**	**4**	**5**	**d_i**
1	M	40	40	M	60	40
2	20	M	0	10	30	0
3	30	0	M	40	0	0
4	0	50	10	M	40	0
5	0	10	30	0	M	0
d_j	0	0	0	0	0	40

Đối với tập chứa (1,4), ta loại tất cả các phần tử dòng 1 và cột 4 và thay thế phần tử (4, 1) bằng ∞. Kết quả, ta nhận được ma trận cấp 4x4.

i j	**1**	**2**	**3**	**5**	**d_i**
2	20	M	0	30	0
3	30	0	M	0	0
4	M	50	10	40	10
5	0	10	30	M	0
d_j	0	0	0	0	10

Cận dưới của tập con (1,4) là 140 + 10 = 150 < 180. Vì cận dưới của tập con (1, 4) nhỏ hơn, nên lộ trình không chứa cạnh (1, 4). Lại tiến hành các bước trên với ma trận 4x4 thu được.

i j	**1**	**2**	**3**	**5**	**d_i**
2	20	M	0(20)	30	20
3	30	0(10)	M	0(30)	0
4	M	40	0(30)	30	30
5	0(30)	10	30	M	10
d_j	20	10	0	30	0

d(1,3) = 20 + 0 = 20; d(2,2) = 0 + 10 = 10; d(2,4) = 0 + 30 = 30; d(3,3) = 30 + 0 = 30; d(4,1) = 10 + 20 = 30; Tổng hằng lớn nhất là (0 + 30) = 30 tương ứng với (3,5), kết quả tập được chia thành hai tập con: chứa (3, 5) và không chứa (3, 5). Cận dưới của tập không chứa (3, 5) là 150 + 30. Đối với tập không chứa (3, 5) ta thay phần tử này bằng ∞.

i j	**1**	**2**	**3**	**5**	**d_i**
2	20	M	0	30	0
3	30	0	M	M	0
4	M	40	0	30	0
5	0	10	30	M	0
d_j	0	0	0	30	30

Đối với tập chứa (3, 5), ta cần loại tất cả các phần tử của dòng 3 và cột 5. Đồng thời, phần tử (5, 3) được thay bằng ∞. Kết quả ta nhận được ma trận 3x3:

i j	**1**	**2**	**3**	**d_i**
2	20	M	0	0
4	M	40	0	0
5	0	10	M	0

d_j	0	10	0	10

Cận dưới của tập con (3,5) là 150 + 10 = 160 < 180. Bởi vì cận dưới của tập chứa (3, 5) nhỏ hơn, nên lộ trình tối ưu sẽ chứa (3, 5). Tiếp tục quá trình này

i j	**1**	**2**	**3**	**d_i**
2	20	M	0(20)	20
4	M	30	0(30)	30
5	0(20)	0(30)	M	0
d_j	20	30	0	0

d(1,3) = 20 + 0 = 20; d(2,3) = 30 + 0 = 30; d(3,1) = 0 + 20 = 20; d(3,2) = 0 + 30 = 30; Tổng lớn nhất (30 + 0) = 30 tương ứng với cạnh (4,3), do đó, tập cạnh lại chia thành: tập con chứa (4, 3) và không chứa (4, 3). Cận dưới của tập không chứa (4, 3) là 160 + 30 = 190. Với tập không chứa cạnh (4, 3), ta thay phần tử (4, 3) bằng ∞.

i j	**1**	**2**	**3**	**d_i**
2	20	M	0	0
4	M	30	M	30
5	0	0	M	0
d_j	0	0	0	30

Trong tập con chứa (4, 3), ta loại dòng 4 và cột 3. Sau đó thay phần tử (3, 4) bằng ∞. Kết quả ta nhận được ma trận 2x2.

i j	**1**	**2**	**d_i**
2	20	M	20
5	0	0	0
d_j	0	0	20

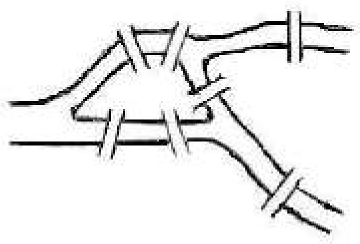
Cận dưới của tập con (4, 3) là 160 + 20 = 180 < 190. Bởi vì cận dưới của tập chứa (4, 3) nhỏ hơn tập không chứa (4, 3) nên lộ trình chứa cạnh (4, 3). Tương ứng với

ma trận này, ta bổ sung thêm hai cạnh (5, 2) và (2, 1). Cuối cùng, lộ trình tối ưu là: (1,4), (4,3), (3,5), (5,2), (2,1). Độ dài của lộ trình tối ưu là 180.

CHƯƠNG 6. LÝ THUYẾT ĐỒ THỊ

6.1. Sơ lược về lý thuyết đồ thị

Lý thuyết đồ thị được nhắc đến lần đầu tiên vào năm 1736, khi Euler khảo sát bài toàn cầu Konigsberg.



Hình 6.1 – Bài toán cầu Konigsberg

Bài toán. Ở thành phố Konigsberg, có một con sông chảy qua và chia vùng đất thành 4 phần. Có bảy cây cầu nối các vùng đất lại với nhau (như trên hình minh họa). Liệu chăng có tồn tại một đường đi xuất phát từ một vùng đất bất kì, đi qua tất cả các cây cầu, mỗi cây cầu chỉ đi qua đúng một lần?

Cho mãi đến năm 1936, bài toán cây cầu Konigsberg mới được nhắc đến trong cuốn sách lí thuyết đồ thị (bằng tiếng Đức và được dịch sang tiếng Anh vào năm 1990). Cũng từ thời điểm này, lí thuyết đồ thị đã có một bước phát triển mạnh mẽ. Nó có một ứng dụng to lớn không chỉ trong toán học, khoa học máy tính cũng như các lĩnh vực có ứng dụng toán học mà còn cả trong các lĩnh vực phi khoa học.

Lí thuyết đồ thị là một ví dụ điển hình cho lớp bài toán NP đầy đủ. Một bài toán gọi là nằm trong lớp P nếu có một thuật toán hữu hiệu để tìm ra nghiệm. Một bài toán gọi là nằm trong lớp NP nếu có một ước đoán hữu hiệu để tìm ra đáp án và một phương pháp hữu hiệu để kiểm tra tính đúng của nó. Nói chung $P \neq NP$. Điều này vẫn còn là một ẩn số trong toán học. Nó là một bài toán lớn trong toán học hiện đại và lý thuyết khoa học máy tính. Nếu ta ước đoán được bài toán trong lớp NP có thể được giải bằng một thuật toán hữu hiệu, thì $P = NP$. Nếu lớp NP hoàn chỉnh nằm trong P thì $P = NP$.

Ghi chú: một thuật toán gọi là hữu hiệu nếu nó có độ phức tạp trong thời gian đa thức (còn được gọi là thuật toán tất định). Một thuật toán gọi là ước đoán hữu hiệu nếu nó có chứa một sự ước đoán bên trong nó (và cũng gọi là thuật toán không tất định).

6.1.1. Các khái niệm về Đồ thị

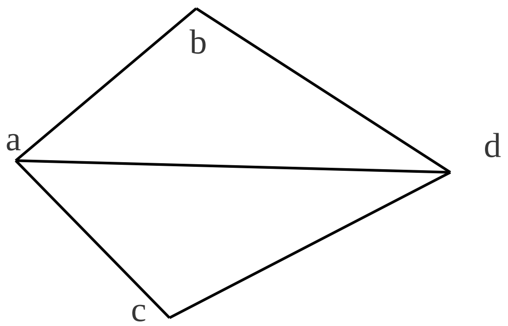
Giả sử V là một tập hữu hạn, ta kí hiệu

$$E(V) = \{\{u, v\} | u, v \in V, u \neq v\}$$

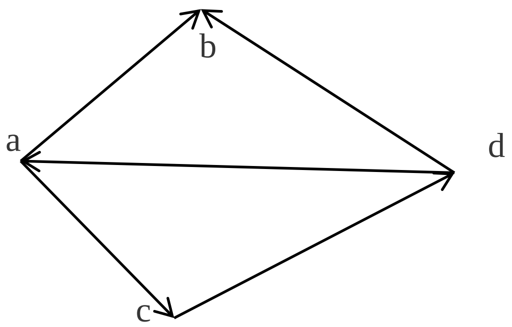
là tập con của V với mỗi cặp phần tử khác nhau.

Định nghĩa. *Một cặp $G = (V, E)$ với $E \subseteq E(V)$ gọi là đồ thị. Các phần tử của V gọi là đỉnh, các phần tử của E gọi là cạnh. Tập các đỉnh của đồ thị G được kí hiệu là V_G hay đơn thuần là V và tập các cạnh của đồ thị G được kí hiệu là E_G hay đơn thuần là E.*

Một cạnh $\{u, v\}$ trong đồ thị gắn kết hai đỉnh u và v của đồ thị, và nó được kí hiệu là uv hay vu (nếu không có sự phân biệt thứ tự u và v). Trong cả hai trường hợp, ta nói đỉnh u liền kề với đỉnh v. Đồ thị không có sự phân biệt thứ tự các đỉnh trong mỗi cạnh gọi là đồ thị vô hướng, ngược lại gọi là đồ thị có hướng.



Hình 6.2 – Đồ thị vô hướng G_1



Hình 6.3 – Đồ thị có hướng G_2

Định nghĩa. *Đối với mỗi đồ thị G, ta kí hiệu*

$$v_G = |V_G|, \epsilon_G = |E_G|$$

Số v_G gọi là cấp của đồ thị G và số ϵ_G gọi là kích thước của đồ thị.

Ví dụ đồ thị G_1 và G_2 ở trên có cấp là 4 và kích thước là 5.

Định nghĩa. *Nếu một cạnh có điểm đầu và điểm cuối trùng nhau thì nó được gọi là khuyên.*

Nếu hai cặp đỉnh bất kì trong đồ thị có nhiều hơn một cặp cạnh nối chúng, thì ta gọi nói là đa đồ thị. Ngược lại, ta gọi là đơn đồ thị.

Một đồ thị có chứa khuyên được gọi là giả đồ thị.

Đồ thị có hướng thường được kí hiệu là D.

Nếu trên mỗi cạnh của đồ thị được gán một giá trị thực thì nó được gọi là đồ thị có trọng số. Trọng số của đồ thị biểu thị cho giá trị, chi phí, lực liên kết....để giữa hai đỉnh liên thuộc trên cạnh này.

Định nghĩa. *Hàm $\alpha: V_G \to K$ là số màu tô cho đỉnh của đồ thị G xác định bởi tập màu K. Hàm $\alpha: E_G \to K$ là số màu tô cho cạnh của đồ thị G xác định bởi tập màu K.*

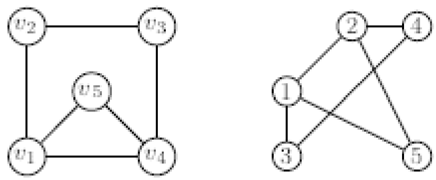
Nếu $K \subseteq R$, thì α gọi là hàm trọng lượng hay hàm khoảng cách.

Định nghĩa. *Hai đồ thị G và H được gọi là đẳng cấu và kí hiệu là $G \cong H$, nếu tồn tại một song ánh $\alpha: V_G \to V_H$ sao cho*

$$uv \in E_G \Leftrightarrow \alpha(u)\alpha(v) \in E_H$$

với mọi $u, v \in G$.

Để chỉ ra hai đồ thị G và H là đẳng cấu, ta cần chỉ ra một song ánh biến đồ thị G thành H hoặc ngược lại.



Hình 6.4 – Hai đồ thị đẳng cấu G và H

Hai đồ thị được cho ở trên là đẳng cấu. Ta dễ dàng chỉ ra song ánh:

$$f: \{v_1 \to 1, v_2 \to 3, v_3 \to 4, v_4 \to 2, v_5 \to 5\}$$

Bài toán đồ thị đẳng cấu. Liệu có tồn tại một thuật toán hiệu quả để kiểm tra hai đồ thị là đẳng cấu hay không đẳng cấu ?

Bảng sau đây liệt kê con số $2^{C_n^2}$ đồ thị có n đỉnh và số lượng đồ thị không đẳng cấu tương ứng.

n	1	2	3	4	5	6	7
Đồ thị	1	2	8	64	1024	32768	2097152
Không Đẳng cấu	1	2	4	11	34	156	1044

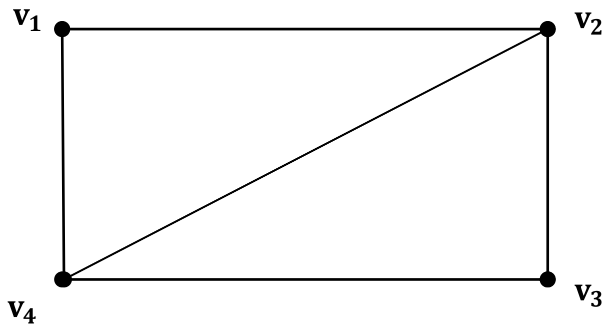
Nói chung không tồn tại một thuật toán hiệu quả để kiểm tra hai đồ thị có là đẳng cấu hay không.

Các dạng biểu diễn của đồ thị. Để biểu diễn một đồ thị, ta có thể thực hiện theo nhiều cách. Chúng ta đã tìm hiểu một cách thức để biểu diễn đồ thị đó là sử dụng hình minh họa. Tuy nhiên, phương pháp này lại không hiệu quả trong lập trình. Trong lập trình, ta thường biểu diễn đồ thị dưới dạng ma trận. Đây là một cách thức hữu hiệu cho việc viết chương trình trên máy tính.

Ma trận kề. Giả sử $V_G = \{v_1, \dots, v_n\}$ là một bộ sắp thứ tự. Ma trận kề của G là một ma trận M cấp $n \times n$ với

$$M_{ij} = \begin{cases} 1, nếu\ v_i v_j \in E_G \\ 0, nếu\ v_i v_j \in E_G \end{cases}$$

Ví dụ:



Hình 6.5 – Biểu diễn đồ thị bằng ma trận kề

Ma trận kề tương ứng với đồ thị có dạng

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Ta cũng cần chú ý rằng, ma trận kề luôn là ma trận đối xứng.

<u>Định lý</u>. *Hai đồ thị G và H gọi là đẳng cấu, nếu và chỉ nếu chúng có cùng ma trận kề. Hơn thế nữa, hai đồ thị đẳng cấu có cùng tập ma trận kề.*

Các đồ thị có thể được biểu diễn dưới dạng tập hợp. Ví dụ $P_X = \{X_1, \dots, X_n\}$ là tập các tập con của tập X, và xác định đồ thị giao G_{P_X} là đồ thị với các đỉnh là $X_1, \dots, X_n$ và cạnh X_iX_j với mọi i, j $(i \neq j)$ và $X_i \cap X_j = \phi$.

<u>Định lý</u>. *Mọi đồ thị là một đồ thị giao của các tập con.*

Chứng minh. Giả sử G là một đồ thị xác định với mọi $v \in G$, và tập

$$X_v = \{\{v, u\} | vu \in E_G\}$$

Khi đó, $X_u \cap X_v = \phi$ nếu và chỉ nếu $uv \in E_G$ ■

Giả sử $s(G)$ là kích thước nhỏ nhất của tập X và G có thể được biểu diễn là đồ thị giao của các tập con của X, khi đó

$$s(G) = \min\{|X| \,\big|\, G \cong G_{P_X} với\ P_X \subseteq 2^X\}$$

$s(G)$ nhỏ như thế nào thì có thể so sánh với v_G (hoặc ϵ_G) của đồ thị ? Người ta đã chỉ ra rằng, đây là một bài toán thuộc lớp NP hoàn chỉnh.

Chúng ta cũng lưu ý rằng, ma trận kề và ma trận liên thuộc đối với đơn đồ thị thực chất là 1. Trong đa đồ thị, ma trận kề sẽ được xác định như sau:

$$M_{n\times n} = (M)_{ij}$$

Trong đó, mỗi phần tử M_{ij} là số cạnh nối đỉnh i với đỉnh j. Nói chung, ma trận kề đối với đồ thị có hướng không phải là ma trận đối xứng.

Nếu ta khảo sát đồ thị có trọng số, thì mỗi phần tử M_{ij} tương ứng với trọng số của cạnh nối đỉnh i với đỉnh j.

<u>Quy ước</u>: giá trị trọng số M_{ij} tương ứng với cạnh (i, j) được quy ước như sau

$$M_{ij} = \begin{cases} x, nếu\ trọng\ số\ của\ cạnh\ (i,j) bằng\ x. \\ 0, nếu\ i = j. \\ +\infty, nếu\ i\ và\ j\ không\ có\ cạnh\ nối. \end{cases}$$

6.1.2. Đồ thị con

Với một đồ thị có kích thước lớn, việc giải quyết bài toán trên đồ thị dạng này là rất khó khăn. Thông thường, người ta sẽ chia các đồ thị này thành các phần nhỏ hơn

dựa trên các đặc tính bộ phận của nó, và tiến hành giải quyết bài toán trên các phần này. Sau đó, kết hợp các thông tin ban đầu để đưa ra kết quả cho đồ thị đã cho. Mỗi phần đồ thị sau khi phân chia như trên gọi là đồ thị con.

Bậc của đỉnh.

<u>Định nghĩa</u>. *Giả sử $v \in G$ là một đỉnh của đồ thị G. Đỉnh liền kề với v là tập hợp*

$$N_G(v) = \{u \in |vu \in E_G\}$$

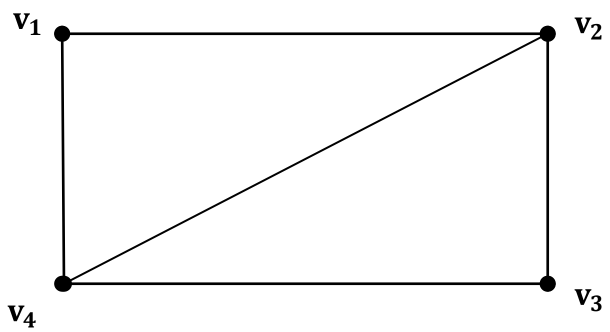
Bậc của v là số đỉnh liền kề với nó hay theo ngôn ngữ tập hợp, nó là lực lượng của tập các đỉnh liền kề với v:

$$deg_G(v) = |N_G(v)|$$

Nếu $deg_G(v) = 0$, thì ta nói v là đỉnh cô lập, nếu $deg_G(v) = 1$ thì ta nói v là đỉnh treo (hay lá) của đồ thị.

Bậc cực tiểu của đồ thị G được kí hiệu là $\delta(G)$, bậc cực đại của đồ thị được kí hiệu là $\Delta(G)$.

Ví dụ. Cho đồ thị sau G:



Hình 6.6 – Bậc của đỉnh trong đồ thị G

Bậc của các đỉnh v_1, v_3 là 2. Bậc của đỉnh v_2, v_4 là 3. Bậc cực tiểu của đồ thị G là 2, bậc cực đại của đồ thị G là 3.

Năm 1736, Euler đã chỉ ra rằng, trong một buổi tiệc, nếu mỗi người đều bắt tay nhau, thì số lượt bắt là một số chẵn.

<u>Bổ đề</u> *<u>(Bổ đề bắt tay)</u>. Với mọi đồ thị G, ta luôn có*

$$\sum_{v \in G} deg_G(v) = 2.\epsilon_G.$$

Hơn nữa, số đỉnh bậc lẻ trong đồ thị G là một số chẵn.

Chứng minh. Mỗi một cạnh của đồ thị đều có hai điểm mút tương ứng. Do đó, tổng bậc của tất cả các đỉnh sẽ gấp đôi số cạnh.

Trong mỗi đồ thị G sẽ có các đỉnh bậc chẵn và các đỉnh bậc lẻ:

$$\sum_{v\in G}\deg_G(v) = \sum_{v\in G,\deg_G(v)=2k}\deg_G(v) + \sum_{v\in G,\deg_G(v)=2k+1}\deg_G(v)$$

Tổng bậc chẵn luôn là một số chẵn và tổng ở vế trái cũng là một số chẵn, nên tổng đỉnh bậc lẻ cũng là một số chẵn. Suy ra số đỉnh bậc lẻ là một số chẵn ■

Định nghĩa. *Một đồ thị $D = (V_D, E_D)$ gọi là đồ thị có hướng nếu mỗi phần tử $uv \in E_D$ là một cặp có thứ tự, có nghĩa là $uv \neq vu$.*

Định nghĩa. *Giả sử D là một đồ thị có hướng. Khi đó, A là*

- *Đồ thị con của nó nếu $V_A \subseteq V_D$ và $E_A \subseteq E_D$,*
- *Đồ thị con thu gọn của nó $A = D[X]$ nếu $V_A = X$ và $E_A = E_D \cap (X \times X)$.*

Định nghĩa. *Bậc vào và bậc ra của đỉnh trong đồ thị có hướng được xác định như sau:*

$$deg_D^+(v) = |\{e \in E_D\ |e = xv\}| - \text{bậc } ra,$$

$$deg_D^-(v) = |\{e \in E_D\ |e = vx\}|\ -\ \text{bậc } vào.$$

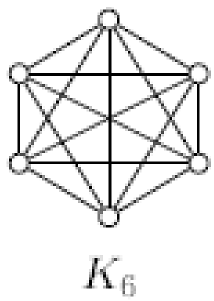
Hay nói cách khác, bậc vào của đỉnh v là tổng số cạnh kết thúc ở v, và bậc ra của đỉnh v là tổng các cạnh xuất phát từ v.

Các dạng đồ thị đặc biệt.

Định nghĩa. *Một đồ thị $G = (V, E)$ là tầm thường nếu nó có một đỉnh, ngược lại gọi là đồ thị không tầm thường.*

Định nghĩa. *Đồ thị $G = K_V$ là một đồ thị hoàn chỉnh trên V nếu mỗi cặp đỉnh trong đồ thị là liền kề nhau.*

Mọi đồ thị hoàn chỉnh cấp n là đẳng cấu với nhau và ta thường kí hiệu là K_n.



Hình 6.7 – Đồ thị hoàn chỉnh K_6

Định nghĩa. *Phần bù của đồ thị G là đồ thị $\overline{G}$ trên V_G, ở đó $E_{\overline{G}} = \{e \in E(V) | e \notin E_G\}$.* *Phần bù của đồ thị hoàn chỉnh là một đồ thị rời rạc.*

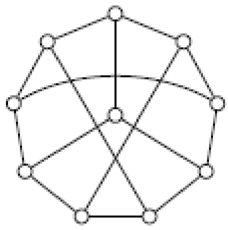
Trong đồ thị rời rạc, tập cạnh là một tập rỗng. Các đồ thị rời rạc kích thước n là đẳng cấu với nhau.

Định nghĩa. *Đồ thị G gọi là chính quy, nếu mọi đỉnh của G có cùng bậc. Nếu bậc này bằng r thì ta gọi G là đồ thị chính quy bậc r hay đơn giản là đồ thị $r -$ chính quy.*

Đồ thị rời rạc là đồ thị chính quy bậc 0 và đồ thị hoàn chỉnh K_n là chính quy $(n-1)$.

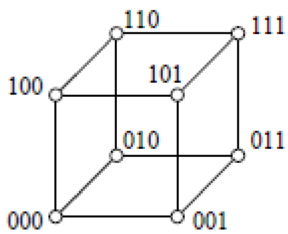
Đồ thị hoàn chỉnh là đồ thị có số cạnh lớn nhất trong tất cả các đồ thị có cùng cấp n: $\epsilon_{K_n} = \frac{n(n-1)}{2}$ và $\epsilon_G \leq \frac{n(n-1)}{2}$ với mọi đồ thị G.

Ví dụ. Đồ thị Petersen là một đồ thị cấp n và nó là đồ thị chính quy bậc 3.



Hình 6.8 – Đồ thị Petersen chính quy bậc 3

Ví dụ. Giả sử $k \geq 1$ là một số nguyên, và giả sử rằng B^k là tập các xâu nhị phân độ dài k. Ví dụ, $B^3 = \{000, 001, 010, 100, 011, 101, 110, 111\}$. Đặt Q_k là khối k - lập phương, trong đó, $V_{Q_k} = B^k$ và $uv \in E_{Q_k}$, nếu và chỉ nếu u và v chỉ khác nhau đúng một vị trí.

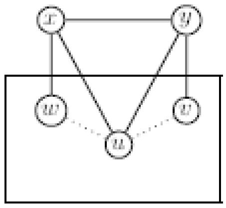


Hình 6.9 – Đồ thị lập phương k – chính quy

Cấp của Q_k là $v_{Q_k} = 2^k$, chính là số xâu kí tự độ dài k. Ta dễ dàng nhận thấy, Q_k là một đồ thị k – chính quy. Theo bổ đề bắt tay, $\epsilon_{Q_k} = k.2^{k-1}$.

Ví dụ. Giả sử $n \geq 4$ là một số chẵn. Chúng ta chỉ ra tồn tại một đồ thị 3 – chính quy G với $v_G = n$. Chú ý rằng, tất cả các đồ thị 3 – chính quy có cấp là một số chẵn.

Nếu n=4, khi đó K_4 là một đồ thị 3 – chính quy. Giả sử đồ thị 3 – chính quy có cấp $2m-2$, và $uv, uw \in E_G$. Đặt $V_H = V_G \cup \{x, y\}$, và $E_H = (E_G\{uv, uw\}) \cup \{ux, xv, uy, yw, xy\}$. Khi đó, H là đồ thị 3 – chính quy cấp $2m$.



Hình 6.10 – Đồ thị 3 – chính quy có cấp là số chẵn

Định nghĩa. *Một đồ thị H là đồ thị con của đồ thị G và kí hiệu là $H \subseteq G$, nếu $V_H \subseteq V_G$ và $E_H \subseteq E_G$.*

Một đồ thị con $H \subseteq G$ gọi là đồ thị con khung của G nếu mọi đỉnh của G là trong H, hay nói cách khác $V_H = V_G$.

Đồ thị con $H \subseteq G$ là một đồ thị con thu gọn, nếu $E_H = E_G \cap E(V_H)$. Trong trường hợp này, H được thu gọn bởi tập đỉnh V_H.

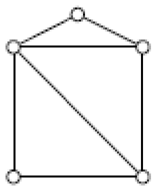
Trong một đồ thị con thu gọn $H \subseteq G$, tập E_H chứa tất cả các đỉnh $e \in E_G$ và $e \in E(V_H)$. Với mỗi tập con khác rỗng $A \subseteq V_G$, có một đồ thị con thu gọn duy nhất

$$G[A] = (A, E_G \cap E(A))$$

Mỗi tập con $F \subseteq E_G$ của các đỉnh có một đồ thị con khung duy nhất

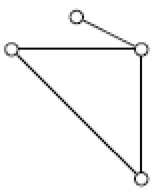
$$G[F] = (V_G, F)$$

Cho đồ thị G.



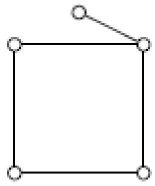
Hình 6.11 – Đồ thị G

Một đồ thị con của G.



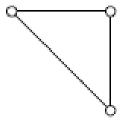
Hình 6.12 – Đồ thị con của đồ thị G

Một đồ thị con khung của G.



Hình 6.13 – Đồ thị con khung của đồ thị G

Một đồ thị con thu gọn của G.



Hình 6.14 – Đồ thị con thu gọn của đồ thị G

Với tập đỉnh $F \subseteq E_G$, đặt

$$G - F = G[E_G - F]$$

là đồ thị con của G nhận được bằng cách bỏ đi cạnh $e \in F$ trong đồ thị G. Hay nói cách khác, đồ thị $G - e$ nhận được bằng cách loại bỏ cạnh $e \in E_G$.

Tương tự, chúng ta viết $G + F$, nếu mỗi cạnh $e \in F$ với $F \subseteq E(V_G)$ được bổ sung vào G.

Với tập con của tập đỉnh $A \subseteq V_G$, đặt $G - A \subseteq G$ là đồ thị con thu gọn bởi $V_G \backslash A$, hay

$$G - A = G[V_G \backslash A]$$

và $G - v$ nhận được từ G bằng cách bỏ đi đỉnh v cùng các các cạnh có nối với v.

Rất nhiều bài toán đồ thị con thu gọn có độ phức tạp lớn. Ví dụ, để tìm một đồ thị con hoàn chỉnh lớn nhất (theo bậc) của đồ thị G cho trước là bài toán thuộc lớn NP.

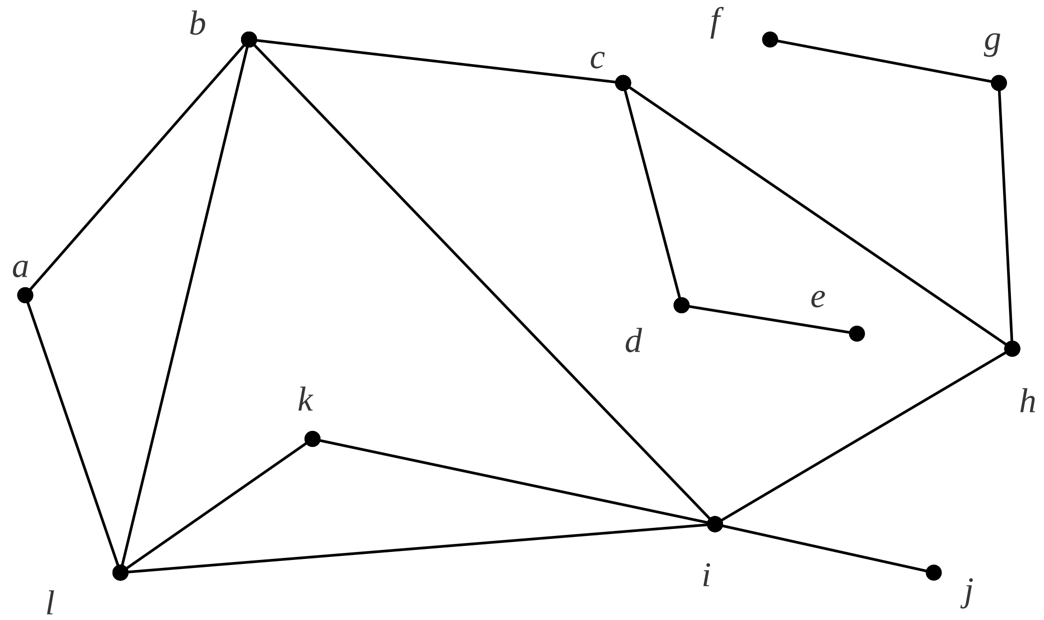
6.1.3. Các phép tìm kiếm trên đồ thị

a) *<u>Tìm kiếm theo chiều sâu</u>*: Thuật toán bắt đầu từ một đỉnh v nào đó. Tiếp tục chọn đỉnh bất kì liền kề với đỉnh v. Quá trình này lại tiếp tục. Nếu tại một bước nào đó, tất cả các đỉnh kế cận với v đều đã được xét duyệt, nhưng vẫn tồn tại một số đỉnh trong đồ thị chưa được xét đến, thì khi đó, chúng ta cần quay lại đỉnh liền trước đỉnh v (tức đỉnh v ở bước lặp trước đó). Thuật toán dừng khi tất cả đỉnh cần tìm được xét duyệt.

b) <u>*Tìm kiếm theo chiều rộng*</u>: Thuật toán bắt đầu từ một đỉnh v nào đó. Giả sử tại một bước lặp nào đó, S là tập đỉnh đã xét, T là tập đỉnh đã được duyệt. Ta đưa vào T các đỉnh của S mà các đỉnh này không có đỉnh kề nào nữa trong G chưa được xét. Tiếp theo, ta thay thế S bằng tập các đỉnh không thuộc T nhưng kề với S. Quá trình này thực hiện cho đến khi S rỗng.

Ví dụ trong đồ thị bên dưới. Ta áp dụng hai thuật toán nêu trên:

- Tìm kiếm theo chiều sâu: *a b c d e h g f i j k l*
- Tìm kiếm theo chiều rộng: *a b l c i k d h j e g f*



Hình 6.15 –Tìm kiếm theo chiều xâu và theo chiều rộng

6.1.4. Hành trình và chu trình

Khi tiến hành giải quyết các bài toán như tìm đường đi ngắn nhất trong mạng, tìm luồng cực đại… chúng ta cần khảo sát các khái niệm liên quan đến đường đi và chu trình. Những dạng bài toán này tương đối khó, hay chính xác hơn là những bài toán không tầm thường, bởi thường có nhiều lựa chọn để tìm ra một nghiệm tối ưu.

<u>Định nghĩa</u>. *Giả sử* $e_i = u_i u_{i+1} \in E_G$ *là các cạnh của G với* $i \in [1, k]$. *Ở đó,* e_i *và* e_{i+1} *là liên thông với mọi* $i \in [1, k-1]$. *Dãy thứ tự*

$$W = e_1 e_2 \dots e_k$$

gọi là đường đi độ dài k từ u_1 *đến* u_{k+1}.

Thông thường, ta viết đường đi như sau:

$$W: u_1 \to u_2 \to \cdots \to u_k \to u_{k+1}$$

hay

$$W: u_1 \overset{k}{\to} u_{k+1}$$

Độ dài của đường đi W được kí hiệu là $|W|$.

Định nghĩa. *Đặt* $W = e_1 e_2 \dots e_k$, $(e_i = u_i u_{i+1})$ *là đường đi.*

W *là khép kín, nếu* $u_1 = u_{k+1}$.

W *là hành trình, nếu* $u_i \neq u_j$ *với mọi* $i \neq j$.

W *là chu trình, nếu nó khép kín và* $u_i \neq u_j$ *với* $i \neq j$ *trừ* $u_1 = u_{k+1}$.

W *là hành trình tầm thường, nếu độ dài của nó là 0. Lộ trình tầm thường không chứa bất kì cạnh nào.*

W^{-1} *gọi là hành trình đảo ngược của* W *nếu* $W^{-1}: e_{k+1} \dots e_1$.

Bổ đề. *Mỗi đường đi* $W: u \overset{*}{\to} v$ *với* $u \neq v$ *chứa một hành trình* $P: u \overset{*}{\to} v$, *mà nó nhận được bằng cách bỏ đi một số cạnh và đỉnh trong* W.

Chứng minh. Đặt $W: u = u_1 \to \cdots \to u_{k+1} = v$. Giả sử $i < j$ và $u_i = u_j$. Nếu không tồn tại một cặp i, j như vậy, thì W sẽ là một hành trình. Ngược lại, trong đường đi $W = W_1 W_2 W_3: u \overset{*}{\to} u_i \overset{*}{\to} u_j \overset{*}{\to} v$ đường đi $U_1 = W_1 W_3: u \overset{*}{\to} u_i = u_j \overset{*}{\to} v$ là đường đi ngắn hơn. Bằng cách lặp đối số này, chúng ta nhận được một dãy $U_1, U_2, \dots, U_m$ của đường đi $u \overset{*}{\to} v$ với $|W| > |U_1| > \cdots > |U_m|$. Khi thủ tục này dừng, ta nhận được một hành trình ■

Định nghĩa. *Nếu tồn tại một đường đi (hay một hành trình) từ* u *vào* v *trong đồ thị* G, *đặt*

$$d_G(u, v) = min\{k, u \overset{k}{\to} v\}$$

là khoảng cách từ u *đến* v. *Nếu không tồn tại đường đi* $u \overset{*}{\to} v$, *đặt* $d_G(u, v) = \infty$. *Một đồ thị* G *là liên thông, nếu* $d_G(u, v) < \infty$ *với mọi* $u, v \in G$; *ngược lại, nó gọi là không liên thông. Số đồ thị con liên thông cực đại của đồ thị* G *gọi là các thành phần liên thông của nó.*

Kí hiệu: $c(G)$ − là số thành phần liên thông của đồ thị G.

Nếu $c(G) = 1$, thì hiển nhiên G là đồ thị liên thông. Đồ thị con $H \subseteq G$ gọi là liên thông cực đại, nếu bất kì một đồ thị con xác định trên tập đỉnh $V_H \cup \{v\}$, với mọi đỉnh $v \notin H$ không là đồ thị liên thông.

Định nghĩa. *Giả sử G^α là một cạnh của đồ thị có trọng số với hàm trọng số $\alpha: E_G \to R$ xác định trên tập cạnh. Với đồ thị con $H \subseteq G$, đặt*

$$\alpha(H) = \sum_{e \in E_H} \alpha(e)$$

là tổng trọng số của H. Trong trường hợp riêng, nếu $P = e_1 e_2 \dots e_k$ là hành trình, thì trọng số của nó là $\alpha(P) = \sum_{i=1}^{k} \alpha(e_i)$. Khoảng cách có trọng số nhỏ nhất giữa hai đỉnh của đồ thị là

$$d_G^\alpha(u, v) = min\left\{\alpha(P) \middle| P: u \xrightarrow{*} v\right\}.$$

Trong các bài toán tối ưu hóa, chúng ta cần tìm một đồ thị con tối ưu $H \subseteq G$ thỏa mãn một số điều kiện. Trong thực tế, chúng ta cần chú ý đến một số tính chất sau:

- Trong các bài toán xây dựng, mạng giao thông…, thì trọng số trên mỗi cạnh của đồ thị có thể là khoảng cách, chi phí vận chuyển, mức đánh giá ưu tiên… trong mạng này.
- Trong một hệ thống các kênh thông tin hoặc kiến trúc máy tính, thì trọng số có thể là tỉ suất không đáng tin hoặc tần số kết nối.
- Trong mô hình phân tử hóa học, thì trọng số tương ứng với lực liên kết phân tử.

Trong những trường hợp này, chúng ta tìm một đồ thị con nối hai đỉnh cho trước hoặc nối tất cả các đỉnh với trọng số nhỏ nhất. Trong một số trường hợp khác, nếu đồ thị biểu diễn một mạng lưới các ống dẫn (dẫn dầu, khí đốt…), thì trọng số có thể là dung tích hoặc sức truyền tải, và khi đó, ta cần tìm đồ thị có trọng số lớn nhất.

Chúng ta khảo sát bài toán cực tiểu. Đối với bài toán này, giả sử cho đồ thị G với hàm trọng số $\alpha: E_G \to N$. Trong trường hợp này, ta gọi $\alpha(uv) -$ là độ dài của hành trình uv.

Bài toán tìm hành trình (đường đi) ngắn nhất: Cho đồ thị liên thông G với hàm trọng lượng $\alpha: E_G \to N$, ta cần tìm $d_G^\alpha(u, v)$ với mọi $u, v \in G$. Hay, có thể phát biểu dưới ngôn ngữ tự nhiên: cho một đồ thị liên thông G, hãy tìm hành trình ngắn nhất giữa hai đỉnh u, v cho trước.

Có nhiều thuật toán để giải bài toán tìm hành trình ngắn nhất. Trong phạm vi của giáo trình này, chúng ta sẽ khảo sát hai thuật toán: Dijkstra và Floyd.

a) *Thuật toán **Dijkstra***.

- Thuật toán Dijkstra chỉ áp dụng cho đồ thị có trọng số dương. Thuật toán Dijkstra là thuật toán có tốc độ thực thi cao nhất trong các thuật toán cùng loại.

Thuật toán. Thuật toán Dijkstra có thể chia làm hai chiều:

+ Chiều thuận: xác định độ dài ngắn nhất từ đỉnh nguồn đến các đỉnh còn lại.

+ Chiều nghịch: xác định các đỉnh nằm trên đường đi ngắn nhất.

Chiều thuận.

+ Đặt $u_0 = u, t(u_0) = 0$ và $t(v) = \infty$ với mọi $v \neq u_0$.

+ Với mỗi $i \in [0, v_G - 1]$: với mỗi $v \notin \{u_1, \dots, u_i\}$, ta thay $t(v)$ bằng $\min\{t(v), t(u_i) + \alpha(u_i v)\}$.

Đặt $u_{i+1} \notin \{u_1, \dots, u_i\}$ là đỉnh bất kì có giá trị nhỏ nhất $t(u_{i+1})$.

+ Kết luận $d_G^\alpha(u, v) = t(v)$.

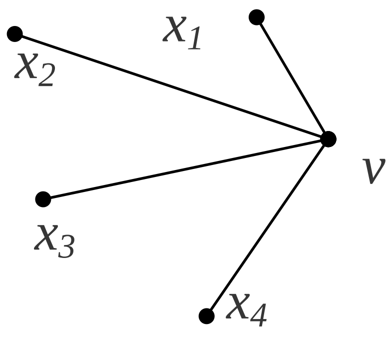
Điều kiện dừng:

+ Nếu yêu cầu tìm đường đi ngắn nhất giữa hai đỉnh bất kì, thì thuật toán dừng khi đường đi ngắn nhất đến đỉnh đích đã được tìm thấy (đỉnh đích đã được đánh dấu).

+ Nếu yêu cầu tìm đường đi ngắn nhất giữa đỉnh nguồn với tất cả các đỉnh còn lại, thì thuật toán dừng khi tất cả các đường đi ngắn nhất đã được tìm thấy.

Chiều nghịch.

Giả sử độ dài của hành trình ngắn nhất đến đỉnh $\boldsymbol{v}$ là $t(v) = m$.



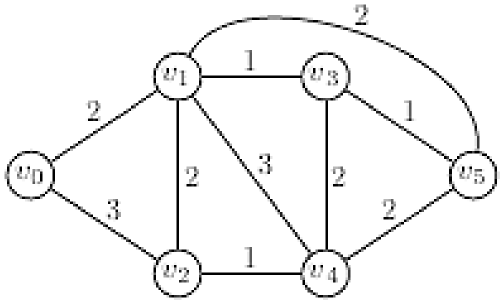
Hình 6.15 – Thuật toán tìm đỉnh liền trước trong hành trình ngắn nhất

Ta tiến hành kiểm tra:

$$t(x_i) + L(x_i, v) = m \qquad (*)$$

Nếu đỉnh x_i nào thỏa mãn đẳng thức trên, thì đỉnh đó nằm ngay trước đỉnh t trong hành trình ngắn nhất.

Ví dụ. Tìm hành trình ngắn nhất từ đỉnh v_0 đến v_5 trong đồ thị sau đây:



Hình 6.16 – Tìm hành trình ngắn nhất bằng thuật toán Dijkstra

Xác định đường đi ngắn nhất đến mỗi đỉnh.

- $u_0 = v_0, t(u_0) = 0, t(u_i) = \infty, \forall i \neq 0.$
- $t(v_1) = \min\{\infty, 2\} = 2, t(v_2) = \min\{\infty, 3\} = 3,$ các đỉnh còn lại bằng ∞. Do đó, chọn $u_1 = v_1$.
- $t(v_2) = \min\{3, t(u_1) + \alpha(u_1 v_2)\} = \min\{3, 4\} = 3, t(v_3) = 2 + 1 = 3, t(v_4) = 2 + 3 = 5, t(v_5) = 2 + 2 = 4$. Do đó, chọn $u_2 = v_3$.
- $t(v_2) = \min\{3, \infty\} = 3, t(v_4) = \min\{5, 3 + 2\} = 5, t(v_5) = \min\{4, 3 + 1\} = 4$. Do đó, chọn $u_3 = v_2$.
- $t(v_4) = \min\{5, 3 + 1\} = 4, t(v_5) = \min\{4, \infty\} = 4$. Do đó, chọn $u_4 = v_4$.
- $t(v_5) = \min\{4, 4 + 1\} = 4$. Thuật toán dừng.

Vậy, các hành trình ngắn nhất đến mỗi đỉnh của đồ thị là

$$t(v_1) = 2, t(v_2) = 3, t(v_3) = 3, t(v_4) = 4, t(v_5) = 4.$$

Xác định các đỉnh nằm trên hành trình ngắn nhất.

Các đỉnh nằm trong hành trình ngắn nhất đến v_5. Các đỉnh liền trước v_5 gồm v_2, v_3, v_4. Trong đó,

$$L(v_1, v_5) = 2, L(v_3, v_5) = 1, L(v_4, v_5) = 2.$$

Kiểm tra điều kiện (*):

$$L(v_1, v_5) + t(v_1) = 2 + 2 = 4 = t(v_5)$$
$$L(v_3, v_5) + t(v_3) = 1 + 3 = 4 = t(v_5)$$

$$L(v_4, v_5) + t(v_4) = 2 + 4 = 6 \neq 4 = t(v_5)$$

Nên đỉnh liền ngay trước v_5 trong hành trình ngắn nhất là v_1 hoặc v_3.

- Nếu chọn v_1, ta làm tương tự như trên và dễ dàng xác định được đỉnh liền trước v_1 trong hành trình ngắn nhất là v_0.
- Nếu chọn v_3, ta cũng thực hiện tương tự và xác định được đỉnh liền trước v_3 là v_1. Liền trước v_1 là v_0.

Như vậy, ta có thể thấy theo thuật toán Dijkstra, ta đã xác định được hai đường đi ngắn nhất từ $v_0 \to v_5$: $v_0 \to v_1 \to v_5$ *hoặc* $v_0 \to v_1 \to v_3 \to v_5$.

Chú ý: Để thuận tiện cho việc biểu diễn thuật toán, ta có thể thực hiện thuật toán Dijkstra theo lược đồ được mô tả bên dưới

$\boldsymbol{t(v_0)}$	$\boldsymbol{t(v_1)}$	$\boldsymbol{t(v_2)}$	$\boldsymbol{t(v_3)}$	$\boldsymbol{t(v_4)}$	$\boldsymbol{t(v_5)}$
0*	∞	∞	∞	∞	∞
	2*	3	∞	∞	∞
		3	3*	5	4
		3*		5	4
				4*	4
					4*

Trong bảng này, giá trị độ dài của hành trình được ghi vào ô tương ứng với đỉnh. Trong mỗi bước lặp, ta chọn ra một đỉnh có hành trình nhỏ nhất và đánh dấu đỉnh đó. Nếu một đỉnh đã được đánh dấu, thì các bước lặp tiếp theo, ta không cần khảo sát đến nó. Giá trị độ dài của hành trình luôn cập nhật trong mỗi bước lặp theo công thức tính $\min\{t(v), t(u_i) + \alpha(u_i v)\}$.

Thuật toán Dijkstra có độ phức tạp $O(|V_G|^2 + |E_G|)$.

b) ***Thuật toán Floyd***. Thuật toán Floyd áp dụng cho cả đồ thị có trọng số âm lẫn trọng số dương. Thuật toán Floyd thường được thực thi trên ma trận trọng số, do đó, nó còn được gọi là thuật toán ma trận.

Thuật toán dựa trên một dãy n ma trận D. Cho đồ thị G, và ma trận trọng số tương ứng $D = (d_{ij})_{n\times n}$. Đặt $d_{ii} = 0, d_{ij}$ – là trọng số của cạnh nối đỉnh $i \to j$, $d_{ij} = \infty$ – nếu không tồn tại cạnh nối đỉnh $i \to j$.

Thuật toán. Thuật toán Floyd được thực hiện thông qua các bước sau:

- Đặt $k = 0 -$ với $0 \leq k \leq n$.
- Tăng $k = k + 1$.
- Với mỗi $i \neq k$ sao cho $d_{ik} \neq \infty$, và với mỗi $j \neq k$ sao cho $d_{kj} \neq \infty$, ta thực hiện tính toán $d_{ij} = \min\{d_{ij}, d_{ik} + d_{kj}\}$.
- Kiểm tra điều kiện dừng. Nếu thỏa mãn thì thuật toán dừng, ngược lại, quay lại bước 2.

Điều kiện dừng.

✓ Nếu tại một bước nào đó $d_{ii} < 0$ mà trong G tồn tại một chu trình có độ dài âm đi qua đỉnh i. Thuật toán dừng. Không tồn tại nghiệm, bài toán không chuẩn mực.

✓ Nếu tất cả các phần tử $d_{ii} > 0$ và $k = n$. Thuật toán dừng. Mỗi phần tử d_{ij} xác định hành trình ngắn nhất từ $i \rightarrow j$.

Thuật toán này cho phép ta xác định độ dài ngắn nhất của hành trình nhưng không chỉ ra các đỉnh nằm trên hành trình này. Để giải quyết vấn đề này, ta cần bổ sung thêm một ma trận Θ_{ij} để tìm các đỉnh nằm trong hành trình ngắn nhất này. Ma trận này sẽ được tính toán song song với ma trận trọng số.

Ban đầu,

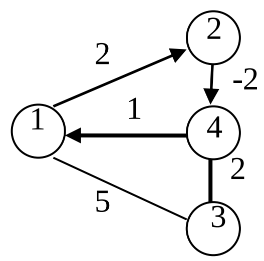
$$\Theta_{ij} = i, \forall i, j$$

Trong quá trình tính toán ma trận trọng số, nếu phần tử nào có giá trị thay đổi, thì phần tử có cùng chỉ số tương ứng trong ma trận Θ cũng sẽ thay đổi theo:

$$\Theta_{ij} = \begin{cases} \Theta_{kj,}\ nếu\ D_{ij}\ thay\ đổi. \\ \Theta_{ij}, nếu\ D_{ij}\ không\ đổi. \end{cases}$$

Mỗi phần tử Θ_{ij} sẽ xác định phần tử liền trước nó.

Ví dụ. Xác định hành trình ngắn nhất từ đỉnh 1 đến đỉnh 3 trong đồ thị sau:



Hình 6.17 – Tìm hành trình ngắn nhất bằng thuật toán Floyd

$$D_0 = \begin{bmatrix} 0 & 2 & 5 & \infty \\ \infty & 0 & \infty & -2 \\ 5 & \infty & 0 & 2 \\ 1 & \infty & 2 & 0 \end{bmatrix} \qquad \Theta_0 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix}$$

$k = 1$

$$D_1 = \begin{bmatrix} 0 & 2 & 5 & \infty \\ \infty & 0 & \infty & -2 \\ 5 & 7 & 0 & 2 \\ 1 & 3 & 2 & 0 \end{bmatrix} \qquad \Theta_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 1 & 3 & 3 \\ 4 & 1 & 4 & 4 \end{bmatrix}$$

$k = 2$

$$D_2 = \begin{bmatrix} 0 & 2 & 5 & 0 \\ \infty & 0 & \infty & -2 \\ 5 & 7 & 0 & 2 \\ 1 & 3 & 2 & 0 \end{bmatrix} \qquad \Theta_2 = \begin{bmatrix} 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 \\ 3 & 1 & 3 & 3 \\ 4 & 1 & 4 & 4 \end{bmatrix}$$

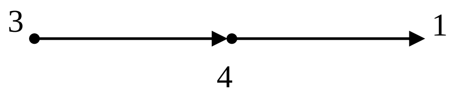
$k = 3$

$$D_3 = \begin{bmatrix} 0 & 2 & 5 & 0 \\ \infty & 0 & \infty & -2 \\ 5 & 7 & 0 & 2 \\ 1 & 3 & 2 & 0 \end{bmatrix} \qquad \Theta_3 = \Theta_2$$

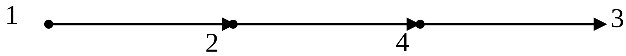
$k = 4$

$$D_4 = \begin{bmatrix} 0 & 2 & 2 & 0 \\ -1 & 0 & 0 & -2 \\ 3 & 5 & 0 & 2 \\ 1 & 3 & 2 & 0 \end{bmatrix} \qquad \Theta_4 = \begin{bmatrix} 1 & 1 & 4 & 2 \\ 4 & 2 & 4 & 2 \\ 4 & 4 & 3 & 3 \\ 4 & 1 & 4 & 4 \end{bmatrix}$$

Trong ma trận D_4 phần tử $D_{13} = 2$, nghĩa là độ dài hành trình ngắn nhất tương ứng là 2. Phần tử tương ứng $\Theta_{13} = 4$, nghĩa là trước đỉnh 3 hành trình phải đi qua đỉnh 4:



Tương tự, ta có $\Theta_{14} = 2$, nghĩa là trước đỉnh 4, hành trình phải đi qua đỉnh 2. Và $\Theta_{12} = 1$, nghĩa là không có đỉnh nào nằm giữa đỉnh 1 và đỉnh 2 trong hành trình ngắn nhất này. Cuối cùng, ta có hành trình ngắn nhất như bên dưới:



Thuật toán Floyd có độ phức tạp là $O(|V_G|^3)$. Trong một số tài liệu khác, thuật toán Floyd còn có tên đầy đủ là thuật toán Floyd – Warshall.

6.2. Đồ thị phân đôi và Cây

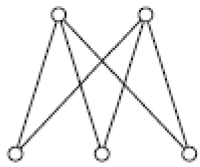
6.2.1. Đồ thị phân đôi và cây

Trong các bài toán tìm đường đi ngắn nhất, chúng ta tìm kiếm một nghiệm tối ưu thỏa mãn điều kiện cho trước. Những nghiệm này thường là các đồ thị con không có chu trình. Những đồ thị liên thông sẽ được gọi là cây, và chúng ta thường sử dụng các thuật toán tìm kiếm dữ liệu. Với các ứng dụng rời rạc, bạn có thể tham khảo giáo trình "Introduction to Algorithm" của Cormen, Leiserson và Rivest do MIT phát hành.

a) *Đồ thị phân đôi.*

Định nghĩa. Một đồ thị G được gọi là đồ thị phân đôi, nếu V_G được chia thành hai tập con không giao nhau X và Y sao cho mỗi cạnh nối trong đồ thị G sẽ nối một đỉnh trong X với một đỉnh trong Y. Khi đó, $(X, Y) -$ gọi là một phân hoạch tập đỉnh V_G. Còn $G(X, Y) -$ gọi là đồ thị phân đôi.

Ví dụ.



Hình 6.17 – Đồ thị phân đôi hoàn chỉnh $K_{2,3}$

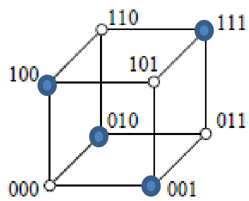
Đồ thị $K_{2,3} -$ là một đồ thị phân đôi hoàn chỉnh.

Chý ý:

- Mọi đồ thị phân đôi hoàn chỉnh $K_{m,k}$ là đẳng cấu với nhau.
- Tập con $X \subseteq V_G$ là ổn định nếu $G[X] -$ là đồ thị rời rạc.

Định lý. *Một đồ thị G là phân đôi, nếu và chỉ nếu V_G phân hoạch thành hai tập con ổn định.*

Ví dụ. Xét đồ thị Q_k. Đồ thị này là đồ thị phân đôi.



Hình 6.18 – Đồ thị Q_k

Nếu ta chia tập đỉnh của đồ thị này thành hai tập:

$$X = \{u|u \text{ có đúng một số lẻ } bit\ 1\}$$

$$Y = \{u|u \text{ có đúng một số chẵn } bit\ 1\}$$

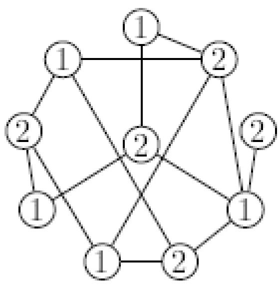
Rõ ràng rằng, hai tập này phân hoạch tập V_{Q_k}, và chúng là ổn định trên Q_k.

Định lý. *Đồ thị G là đồ thị phân đôi nếu và chỉ nếu nó không có chu trình lẻ.*

Thuật toán kiểm tra đồ thị phân đôi. Việc kiểm tra một đồ thị là phân đôi là một bài toán đơn giản. Ta có thể kiểm tra bằng thuật toán tô hai màu. Giả sử gọi hai màu tô cho mỗi đỉnh là màu 1 và màu 2.

- Chọn một đỉnh bất kì và tô màu cho nó là màu 1.
- Chọn tất cả các đỉnh liền kề với đỉnh này, ta tô màu cho nó là màu 2.
- Tiếp tục bước 2.
- Nếu tất cả các đỉnh đều được tô bằng đúng hai màu thì thuật toán dừng. Đồ thị đã cho là đồ thị phân đôi. Ngược lại, nếu một đỉnh nào đó nhận được hai màu tô cùng lúc, thuật toán dừng. Đồ thị đã cho không phải là đồ thị phân đôi.

Ví dụ. Đồ thị sau đây là phân đôi.



Hình 6.19 – Thuật toán tìm đồ thị phân đôi

Định lý Erdos. *Mọi đồ thị G có một đồ thị con phân đôi* $H \subseteq G$ *mà* $\epsilon_H \geq \frac{1}{2}\epsilon_G$.

Chứng minh. Đặt $V_G = X \cup Y$ là một phân hoạch tập đỉnh của G sao cho số cạnh nối X và Y là lớn nhất. Kí hiệu

$$F = E_G \cap \{uv | u \in X, v \in Y\},$$

và đặt $H = G[F]$. Hiển nhiên, H là một đồ thị con khung và nó là phân đôi. Theo điều kiện cực đại, ta có

$$d_H(v) \geq \frac{1}{2} d_G(v),$$

Cho nên,

$$\epsilon_H = \frac{1}{2} \sum_{v \in H} d_H(v) \geq \frac{1}{2} \sum_{v \in G} \frac{1}{2} d_G(v) = \frac{1}{2} \epsilon_G \ \blacksquare$$

b) <u>*Cầu*</u>.

<u>Định nghĩa</u>. *Một cạnh $e \in E_G$ là cầu của đồ thị G, nếu $G - e$ có nhiều hơn G một thành phần liên thông và $c(G - e) > c(G)$.*

<u>Định lý</u>. *Một cạnh $e \in E_G$ là cầu nếu và chỉ nếu e không thuộc vào một chu trình nào của G.*

Chứng minh.

($\Longrightarrow$) Nếu có một chu trình trong G chứa e, thì có một chu trình $C = eP: u \to v \xrightarrow{*} u$, ở đó $P: v \xrightarrow{*} u$ là hành trình trong $G - e$, và e không phải là cầu.

($\Longleftarrow$) Giả sử rằng $e = uv$ không phải là cầu. Vì vậy, u và v nằm trong cùng một thành phần liên thông của $G - e$. Nếu $P: v \xrightarrow{*} u$ là hành trình trong $G - e$, khi đó $eP: u \to v \xrightarrow{*} u$ là chu trình trong G và nó chứa e $\blacksquare$

<u>Bổ đề</u>. *Giả sử e là cầu trong một đồ thị liên thông G.*

(i) $-$ *Khi đó $c(G - e) = 2$.*

(ii) $-$ *Nếu H là một thành phần liên thông của $G - e$. Nếu $f \in E_H$ là một cầu của H, khi đó f là một cầu của G.*

c) <u>*Cây*</u>.

<u>Định nghĩa</u>. *Một đồ thị gọi là không tuần hoàn, nếu nó không có chu trình. Một đồ thị không tuần hoàn gọi là rừng. Một đồ thị không tuần hoàn liên thông gọi là cây.*

<u>Hệ quả</u>. *Một đồ thị liên thông là một cây nếu và chỉ nếu tất cả các cạnh của nó là cầu.*

<u>Định lý</u>. *Các khẳng định sau đây là tương đương*

$(i) -$ *T là cây.*

$(ii) -$ *Bất kì hai đỉnh liền kề trong T chỉ có duy nhất một hành trình.*

$(iii) -$ *T không chứa chu trình và* $\epsilon_T = v_T - 1$.

6.2.2. Cây khung của đồ thị.

Định nghĩa. *Một đồ thị khung là cây được gọi là cây khung.*

Định lý. *Mỗi đồ thị liên thông đều có một cây khung.*

Hệ quả 1. *Mỗi đồ thị liên thông G đều có* $\epsilon_G \geq v_G - 1$. *Hơn nữa, một đồ thị liên thông G là cây nếu và chỉ nếu* $\epsilon_G = v_G - 1$.

Hệ quả 2. *Với mỗi cây T mà* $v_T \geq 2$ *có ít nhất 2 lá.*

Bài toán cây khung nhỏ nhất. Bài toán tìm cây khung có tổng trọng số nhỏ nhất là một bài toán có nhiều ứng dụng trong thực tiễn.

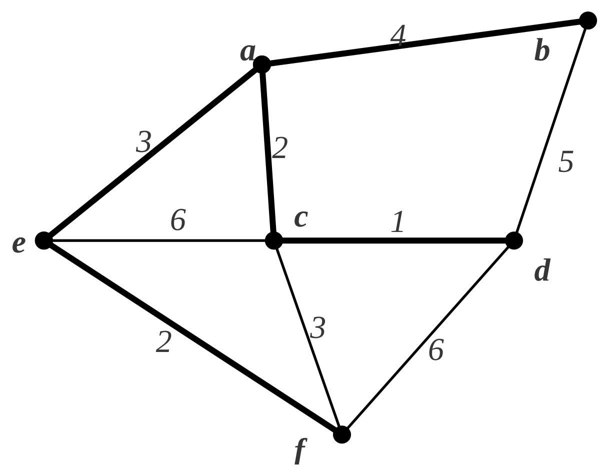
Để giải quyết bài toán này, chúng ta sẽ khảo sát hai thuật toán: Kruskal và Prim.

a) *Thuật toán **Prim***.

Thuật toán.

- Chọn một cạnh bất kì trong đồ thị có trọng số nhỏ nhất và bổ sung vào cây khung.
- Duyệt tất cả những cạnh liền kề với cạnh trên và chọn ra cạnh có trọng số nhỏ nhất để bổ sung vào cây khung sao cho không tạo ra chu trình. Lặp lại quá trình này cho đến khi cây khung có $|V_G| - 1$ cạnh.

Ví dụ.



Hình 6.20 – Thuật toán Prim

Các bước thực hiện thuật toán được liệt kê theo bảng sau đây:

Bước	Cạnh	Trọng số
1	{c, d}	1
2	{c, a}	2
3	{a, e}	3
4	{e, f}	2
5	{a, b}	4
	Trọng số tối ưu:	12

Trong một số giáo trình, thuật toán Prim có có cách biểu diễn khác: không nhất thiết phải chọn cạnh có trọng số nhỏ nhất trong bước lặp đầu tiên, mà ta có thể chọn ra một đỉnh bất kì, và tiếp tục chọn cạnh có trọng số nhỏ nhất liên thuộc với đỉnh đó. Các bước tiếp theo không có gì thay đổi.

b) *Thuật toán* ***Kruskal***.

Thuật toán.

- Chọn một cạnh bất kì trong đồ thị có trọng số nhỏ nhất và bổ sung vào cây khung.
- Khác với thuật toán Prim, cần duyệt các cạnh liền kề với các cạnh đã bổ sung vào cây khung, thuật toán Kruskal không yêu cầu tính chất này. Ta chỉ cần chọn các cạnh có trọng số nhỏ nhất và đảm bảo các cạnh đó không tạo thành một chu trình.

Ví dụ.

Áp dụng thuật toán Kruskal cho đồ thị ở trên, ta có kết quả như sau:

Bước	Cạnh	Trọng số
1	{c, d}	1
2	{c, a}	2
3	{e, f}	2
4	{a, e}	3
5	{a, b}	4
	Trọng số tối ưu:	12

∴∴∴ ***Chý ý:*** Nếu tại một bước lặp nào đó, có hai cạnh có trọng số bằng nhau, thì ta ưu tiên theo thứ tự bảng chữ cái.

6.2.3. Các phép duyệt cây

a) *Duyệt cây theo thứ tự trước*

- Duyệt gốc.
- Duyệt nhánh bên trái sang theo thứ tự trước.

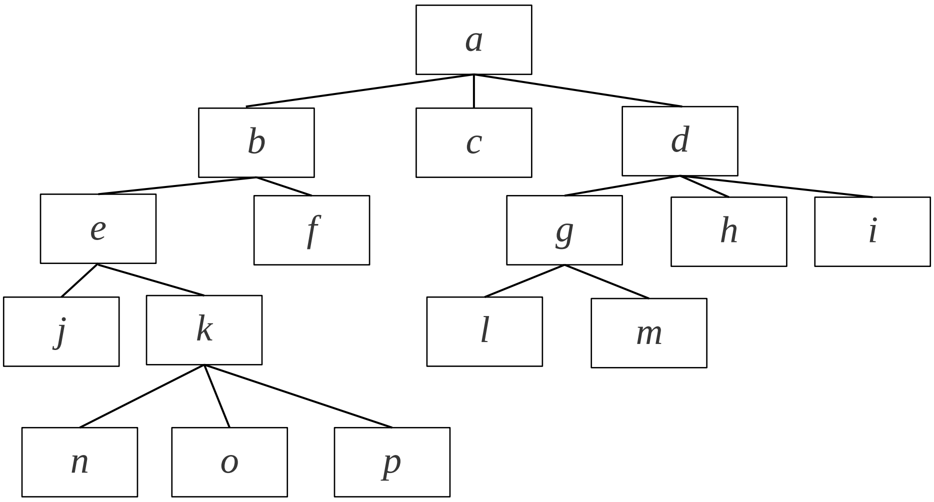
b) *Duyệt cây theo thứ tự giữa*

- Duyệt nhánh bên trái theo thứ tự giữa.
- Duyệt gốc.
- Duyệt các nhánh còn lại theo thứ tự giữa.

c) *Duyệt cây theo thứ tự sau*

- Duyệt các nhánh theo thứ tự sau từ trái sang.
- Duyệt gốc.

Ví dụ. Cho cây như sau:



Hình 6.21 – Các phép duyệt cây

- Duyệt cây theo thứ tự trước: *abejknopfcdglmhi.*
- Duyệt cây theo thứ tự giữa: *jenkopbfaclgmdhi.*
- Duyệt cây theo thứ tự sau: *jnopkefbclmghida.*

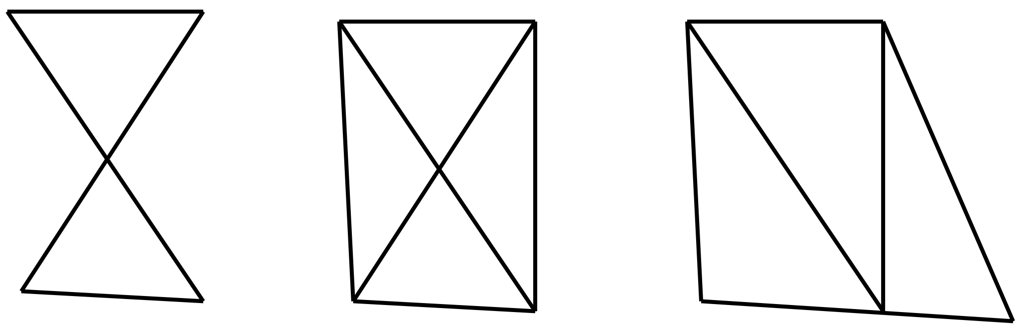
6.3. Đồ thị Euler và Đồ thị Hamilton

6.3.1. Đồ thị Euler

Định nghĩa.

- *Chu trình đơn chứa tất cả các cạnh của đồ thị gọi là chu trình Euler.*
- *Đường đi đơn chứa tất cả các cạnh của đồ thị gọi là đường đi Euler.*
- *Đồ thị được gọi là đồ thị Euler nếu nó chứa một chu trình Euler.*
- *Đồ thị được gọi là nửa Euler nếu nó chứa đường đi Euler.*

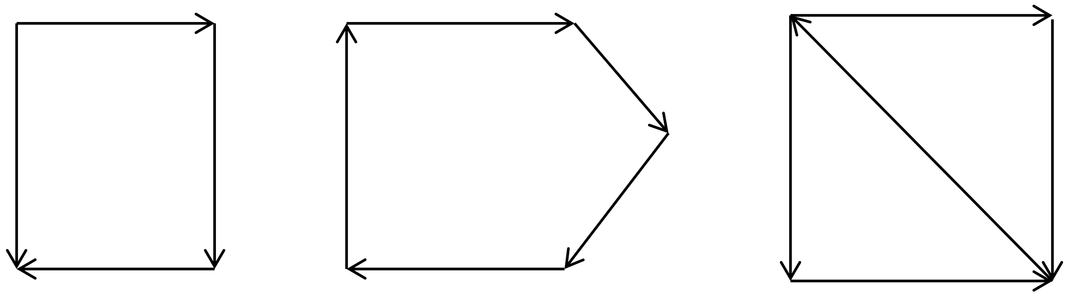
Ví dụ.



$\boldsymbol{G_1}$ $\boldsymbol{G_2}$ $\boldsymbol{G_3}$

Hình 6.22 – Đồ thị vô hướng và tính Euler

Đồ thị $\boldsymbol{G_1}$ là đồ thị Euler. Đồ thị $\boldsymbol{G_3}$ là đồ thị nửa Euler.



H_1 H_2 H_3

Hình 6.23 – Đồ thị có hướng và tính Euler

Đồ thị G_1 là đồ thị Euler. Đồ thị G_3 là đồ thị nửa Euler.

Định lý. *Một đa đồ thị vô hướng liên thông là đồ thị Euler khi và chỉ khi mỗi đỉnh của nó đều có bậc chẵn.*

Hệ quả. *Một đa đồ thị vô hướng liên thông có đường đi Euler (và không có chu trình Euler) khi và chỉ khi nếu có đúng hai đỉnh bậc lẻ.*

Định lý. *Một đa đồ thị có hướng liên thông yếu là đồ thị Euler khi và chỉ khi mọi đỉnh đều có bậc vào bằng bậc ra.*

Định lý. *Một đa đồ thị có hướng liên thông yếu là đồ thị nửa Euler khi và chỉ khi tồn tại đúng hai đỉnh x và y sao cho:*

$$\deg^+(x) = \deg^-(x) + 1;\ \ \deg^-(y) = \deg^+(y) + 1; \deg^+(v) = \deg^-(v).$$

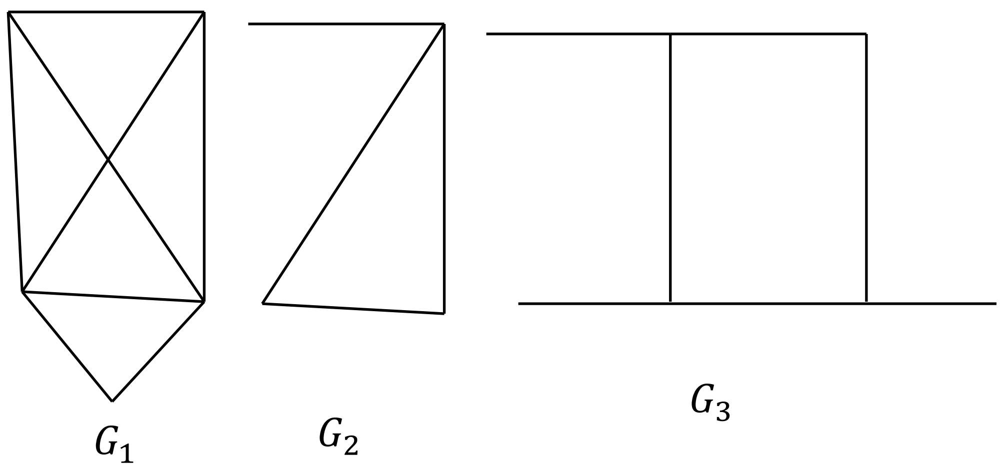
Trong đó, $\forall v \in V, v \neq x, v \neq y$.

6.3.2. Đồ thị Hamilton

Định nghĩa.

- *Chu trình đơn chứa tất cả các đỉnh của đồ thị gọi là chu trình Hamilton.*
- *Đường đi đơn chứa tất cả các đỉnh của đồ thị gọi là đường đi Hamilton.*
- *Đồ thị được gọi là đồ thị Hamilton nếu nó chứa một chu trình Hamilton.*

Ví dụ.



Hình 6.24 – Đồ thị vô hướng và tính Hamilton

Đồ thị G_1 có chu trình Hamilton nên nó là đồ thị Hamilton. Đồ thị G_2 có đường đi Hamilton. Đồ thị G_3 không có chu trình Hamilton lẫn đường đi Hamilton.

Bổ đề. *Nếu G là một đồ thị Hamilton, khi đó, với mỗi tập con khác rỗng* $S \subseteq V_G$*, ta có*

$$c(G - S) \leq |S|.$$

Định lý ORE. *Cho G là một đồ thị có cấp* $v_G \geq 3$*, và giả sử* $\forall u, v \in G$*, trong đó* u, v *là hai đỉnh không liền kề trong G và thỏa mãn*

$$d_G(u) + d_G(v) \geq v_G.$$

Khi đó, G là đồ thị Hamilton.

Hệ quả. *Giả sử G là một đơn đồ thị liên thông có cấp* $v_G \geq 3$. *Khi đó G là đồ thị Hamilton nếu bậc của mỗi đỉnh ít nhất bằng* $\frac{v_G}{2}$.

Định lý. *Mọi đồ thị có hướng đầy đủ là đồ thị nửa Hamilton.*

Định lý Dirac. *Đơn đồ thị liên thông n đỉnh. G là đồ thị Hamilton nếu bậc của mỗi đỉnh ít nhất bằng* $\left\lfloor\frac{n}{2}\right\rfloor$.

Hệ quả. *Nếu G là đơn đồ thị có n đỉnh và mọi đỉnh của G đều có bậc không nhỏ hơn* $\left\lfloor\frac{n-1}{2}\right\rfloor$ *thì G là đồ thị nửa Hamilton.*

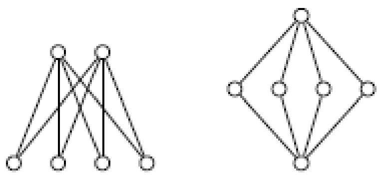
Định lý. *Nếu G là đồ thị phân đôi với hai tập đỉnh là* V_1, V_2 *có số đỉnh cùng bằng* $n, n \geq 2$ *và bậc của mỗi đỉnh lớn hơn* $\frac{n}{2}$ *thì G là một đồ thị Hamilton.*

Định lý. *Đồ thị đầy đủ* K_n *với n lẻ và* $n \geq 3$ *có đúng* $\frac{n-1}{2}$ *chu trình Hamilton phân biệt.*

6.4. Đồ thị phẳng

Định nghĩa. *Một đồ thị G là đồ thị phẳng, nếu nó có thể biểu diễn trên một mặt phẳng mà các cạnh của nó không giao nhau.*

Ví dụ. Đồ thị phân đôi hoàn chỉnh $K_{2,4}$ là đồ thị phẳng.

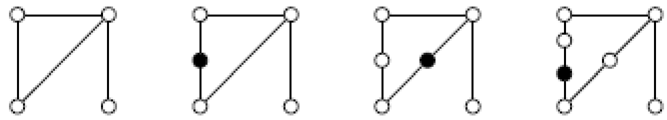


Hình 6.25 – Đồ thị phẳng

Định nghĩa. *Một cạnh* $e = uv \in E_G$ *được gọi là bị chia nhỏ (hay phép phân chia sơ cấp cạnh e), nếu nó được thay thế bằng hành trình* $u \to x \to v$.

- *Một đồ thị H được gọi là phôi của G nếu nó nhận được từ G sau một dãy phép phân chia sơ cấp.*
- *Hai đồ thị được gọi là đồng phôi nếu chúng nhận được từ cùng một đồ thị sau một số phép phân chia sơ cấp.*

Ví dụ.



Hình 6.26 – Phép phân chia sơ cấp

Bổ đề. Một đồ thị là phẳng nếu và chỉ nếu các đồ thị phân nhỏ của nó là phẳng.

Đồ thị phẳng cực đại

Định lý. *Nếu G là một đồ thị phẳng với bậc* $v_G \geq 3$, *khi đó* $\epsilon_G \leq 3v_G - 6$. *Hơn nữa, nếu G không chứa một đồ thị* C_3, *thì* $\epsilon_G \leq 2v_G - 4$.

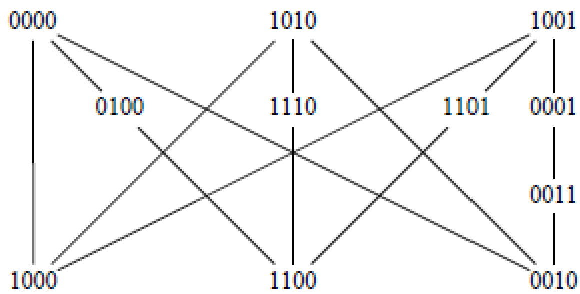
Định lý Heawood. *Nếu G là một đồ thị phẳng, thì* $\delta(G) \leq 5$.

Định nghĩa. *Một đồ thị phẳng G gọi là cực đại, nếu* $G + e$ *không phải là một đồ thị phẳng với mọi* $e \notin E_G$.

Định lý. *Đồ thị* K_5 *và* $K_{3,3}$ *không là đồ thị phẳng.*

Định lý Kuratowski. *Một đồ thị là phẳng khi và chỉ khi nó không chứa một đồ thị con đồng phôi với* K_5 *hoặc* $K_{3,3}$.

Ví dụ. Đồ thị Q_k là đồ thị phẳng chỉ với $k = 1, 2, 3$. Đồ thị Q_4 không là đồ thị phẳng vì nó chứa một đồ thị con đồng phôi với $K_{3,3}$. Với đồ thị $Q_k, k \geq 4$ luôn chứa một đồ thị con Q_4, nên hiển nhiên chúng cũng không là đồ thị phẳng.



Hình 6.27 – Đồ thị phân đôi

BÀI TẬP TỔNG HỢP

PHẦN 1. BÀI TOÁN ĐẾM

1) Trong tổng số 2504 sinh viên của một khoa công nghệ thông tin, có 1876 theo học môn ngôn ngữ lập trình Pascal, 999 học môn ngôn ngữ Fortran và 345 học ngôn ngữ C. Ngoài ra còn biết 876 sinh viên học cả Pascal và Fortran, 232 học cả Fortran và C, 290 học cả Pascal và C. Nếu 189 sinh viên học cả 3 môn Pascal, Fortran và C thì trong trường hợp đó có bao nhiêu sinh viên không học môn nào trong 3 môn ngôn ngữ lập trình kể trên.

2) Một cuộc họp gồm 12 người tham dự để bàn về 3 vấn đề. Có 8 người phát biểu về vấn đề I, 5 người phát biểu về vấn đề II và 7 người phát biểu về vấn đề III. Ngoài ra, có đúng 1 người không phát biểu vấn đề nào. Hỏi nhiều lắm là có bao nhiêu người phát biểu cả 3 vấn đề.

3) Chỉ ra rằng có ít nhất 4 người trong số 25 triệu người có cùng tên họ viết tắt bằng 3 chữ cái sinh cùng ngày trong năm (không nhất thiết trong cùng một năm).

4) Một tay đô vật tham gia thi đấu giành chức vô địch trong 75 giờ. Mỗi giờ anh ta có ít nhất một trận đấu, nhưng toàn bộ anh ta có không quá 125 trận. Chứng tỏ rằng có những giờ liên tiếp anh ta đã đấu đúng 24 trận.

5) Cho n là số nguyên dương bất kỳ. Chứng minh rằng luôn lấy ra được từ n số đã cho một số số hạng thích hợp sao cho tổng của chúng chia hết cho n.

6) Trong một cuộc lấy ý kiến về 7 vấn đề, người được hỏi ghi vào một phiếu trả lời sẵn bằng cách để nguyên hoặc phủ định các câu trả lời tương ứng với 7 vấn đề đã nêu. Chứng minh rằng với 1153 người được hỏi luôn tìm được 10 người trả lời giống hệt nhau.

7) Có 17 nhà bác học viết thư cho nhau trao đổi 3 vấn đề. Chứng minh rằng luôn tìm được 3 người cùng trao đổi một vấn đề.

8) Trong kỳ thi kết thúc học phần toán học rời rạc có 10 câu hỏi. Có bao nhiêu cách gán điểm cho các câu hỏi nếu tổng số điểm bằng 100 và mỗi câu ít nhất được 5 điểm.

9) Phương trình $x_1 + x_2 + x_3 + x_4 + x_5 = 21$ có bao nhiêu nghiệm nguyên không âm?

10) Có bao nhiêu xâu khác nhau có thể lập được từ các chữ cái trong từ *MISSISSIPI*, yêu cầu phải dùng tất cả các chữ?

11) Một giáo sư cất bộ sưu tập gồm 40 số báo toán học vào 4 chiếc ngăn tủ, mỗi ngăn đựng 10 số. Có bao nhiêu cách có thể cất các tờ báo vào các ngăn nếu:

a) Mỗi ngăn được đánh số sao cho có thể phân biệt được;

b) Các ngăn là giống hệt nhau?

12) Tìm hệ thức truy hồi cho số mất thứ tự D_n.

13) Tìm hệ thức truy hồi cho số các xâu nhị phân chứa xâu 01.

14) Tìm hệ thức truy hồi cho số cách đi lên n bậc thang nếu một người có thể bước một, hai hoặc ba bậc một lần.

15) Giải các bài toán sau:

a) Tìm hệ thức truy hồi mà R_n thoả mãn, trong đó R_n là số miền của mặt phẳng bị phân chia bởi n đường thẳng nếu không có hai đường nào song song và không có 3 đường nào cùng đi qua một điểm.
b) Tính R_n bằng phương pháp lặp.
16) Tìm nghiệm của hệ thức truy hồi $a_n = 2a_{n-1} + 5a_{n-2} - 6a_{n-3}$ với $a_0 = 7$, $a_1 = -4$, $a_2 = 8$.

PHẦN 2. CÁC KHÁI NIỆM CƠ BẢN VỀ ĐỒ THỊ

1) Cho G là đồ thị có v đỉnh và e cạnh, còn M, m tương ứng là bậc lớn nhất và nhỏ nhất của các đỉnh của G. Chứng tỏ rằng

$$m \le \frac{2e}{v} \le M.$$

2) Chứng minh rằng nếu G là đơn đồ thị phân đôi có v đỉnh và e cạnh, khi đó

$$e \le v^2/4.$$

3) Trongmột phương án mạng kiểu lưới kết nối $n=m^2$ bộ xử lý song song, bộ xử lý P(i,j) được kết nối với 4 bộ xử lý (P(i±1) mod m, j), P(i, (j±1) mod m), sao cho các kết nối bao xung quanh các cạnh của lưới. Hãy vẽ mạng kiểu lưới có 16 bộ xử lý theo phương án này.
4) Hãy vẽ các đồ thị vô hướng được biểu diễn bởi ma trận liền kề sau:

$$\text{a)}\begin{pmatrix}1&2&3\\2&0&4\\3&4&0\end{pmatrix},\ \text{b)}\begin{pmatrix}1&2&0&1\\2&0&3&0\\0&3&1&1\\1&0&1&0\end{pmatrix},\ \text{c)}\begin{pmatrix}0&1&3&0&4\\1&2&1&3&0\\3&1&1&0&1\\0&3&0&0&2\\4&0&1&2&3\end{pmatrix}.$$

5) Nêu ý nghĩa của tổng các phần tử trên một hàng (t.ư. cột) của một ma trận liền kề đối với một đồ thị vô hướng ? Đối với đồ thị có hướng ?
6) Tìm ma trận liền kề cho các đồ thị sau:

a) K_n , b) C_n, c) W_n , d) $K_{m,n}$, e) Q_n.

7) Có bao nhiêu đơn đồ thị không đẳng cấu với n đỉnh khi:

a) n=2, b) n=3, c) n=4.

8) Hai đơn đồ thị với ma trận liền kề sau đây có là đẳng cấu không?

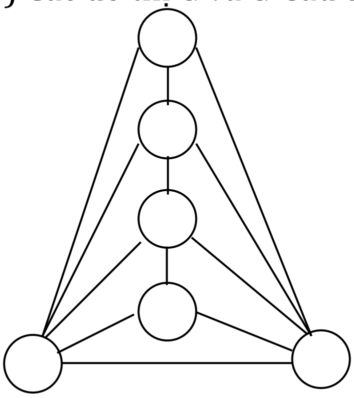
$$\begin{pmatrix}0&1&0&1\\1&0&0&1\\0&0&0&1\\1&1&1&0\end{pmatrix},\ \begin{pmatrix}0&1&1&1\\1&0&0&1\\1&0&0&1\\1&1&1&0\end{pmatrix}.$$

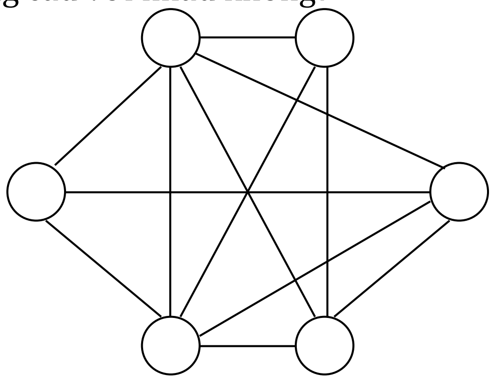
9) Hai đơn đồ thị với ma trận liền kề sau đây có là đẳng cấu không?

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

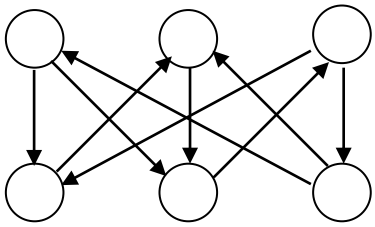
10) Các đồ thị G và G’ sau có đẳng cấu với nhau không?

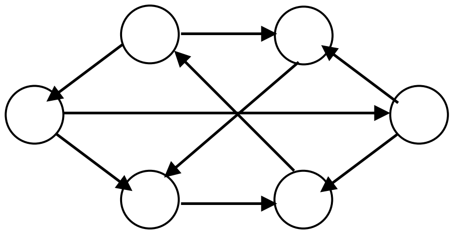
a)





b)





11) Cho V={2,3,4,5,6,7,8} và E là tập hợp các cặp phần tử (u,v) của V sao cho u<v và u,v nguyên tố cùng nhau. Hãy vẽ đồ thị có hướng G=(V,E). Tìm số các đường đi phân biệt độ dài 3 từ đỉnh 2 tới đỉnh 8.

12) Hãy tìm số đường đi độ dài n giữa hai đỉnh liền kề (t.ư. không liền kề) tùy ý trong $K_{3,3}$ với mỗi giá trị của n sau:

a) n=2, b) n=3, c) n=4, d) n=5.

13) Một cuộc họp có ít nhất ba đại biểu đến dự. Mỗi người quen ít nhất hai đại biểu khác. Chứng minh rằng có thể xếp được một số đại biểu ngồi xung quanh một bàn tròn, để mỗi người ngồi giữa hai người mà đại biểu đó quen.

14) Một lớp học có ít nhất 4 sinh viên. Mỗi sinh viên thân với ít nhất 3 sinh viên khác. Chứng minh rằng có thể xếp một số chẵn sinh viên ngồi quanh một cái bàn tròn để mỗi sinh viên ngồi giữa hai sinh viên mà họ thân.

15) Trong một cuộc họp có đúng hai đại biểu không quen nhau và mỗi đại biểu này có một số lẻ người quen đến dự. Chứng minh rằng luôn luôn có thể xếp một số đại biểu ngồi chen giữa hai đại biểu nói trên, để mỗi người ngồi giữa hai người mà anh ta quen.

16) Một thành phố có n (n ≥ 2) nút giao thông và hai nút giao thông bất kỳ đều có số đầu mối đường ngầm tới một trong các nút giao thông này đều không nhỏ hơn n. Chứng minh rằng từ một nút giao thông tuỳ ý ta có thể đi đến một nút giao thông bất kỳ khác bằng đường ngầm.

PHẦN 3. ĐỒ THỊ EULER VÀ HAMILTON

1) Với giá trị nào của n các đồ thị sau đây có chu trình Euler ?

a) K_n, b) C_n, c) W_n, d) Q_n.

2) Với giá trị nào của m và n các đồ thị phân đôi đầy đủ $K_{m,n}$ có:

a) chu trình Euler ? b) đường đi Euler ?

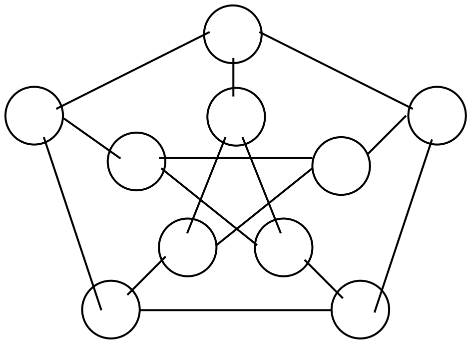
3) Với giá trị nào của m và n các đồ thị phân đôi đầy đủ $K_{m,n}$ có chu trình Hamilton ?

4) Chứng minh rằng đồ thị lập phương Q_n là một đồ thị Hamilton. Vẽ cây liệt kê tất cả các chu trình Hamilton của đồ thị lập phương Q_3.

5) Trong một cuộc họp có 15 người mỗi ngày ngồi với nhau quanh một bàn tròn một lần. Hỏi có bao nhiêu cách sắp xếp sao cho mỗi lần ngồi họp, mỗi người có hai người bên cạnh là bạn mới, và sắp xếp như thế nào ?

6) Hiệu trưởng mời 2n ($n \geq 2$) sinh viên giỏi đến dự tiệc. Mỗi sinh viên giỏi quen ít nhất n sinh viên giỏi khác đến dự tiệc. Chứng minh rằng luôn luôn có thể xếp tất cả các sinh viên giỏi ngồi xung quanh một bàn tròn, để mỗi người ngồi giữa hai người mà sinh viên đó quen.

7) Đồ thị cho trong hình sau gọi là đồ thị Peterson P.

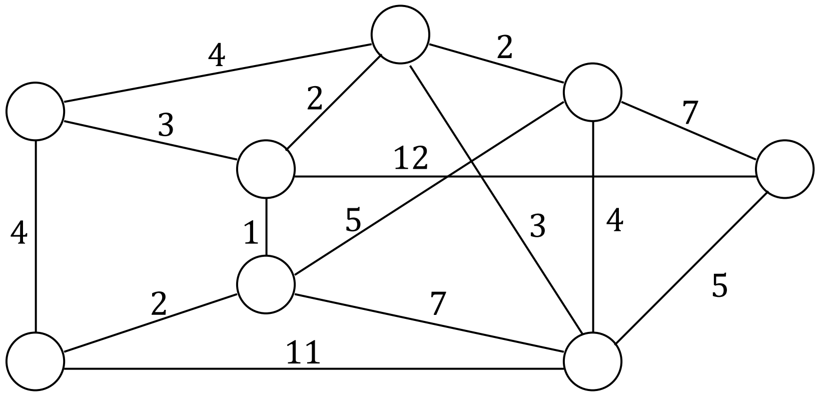


a) Tìm một đường đi Hamilton trong P.

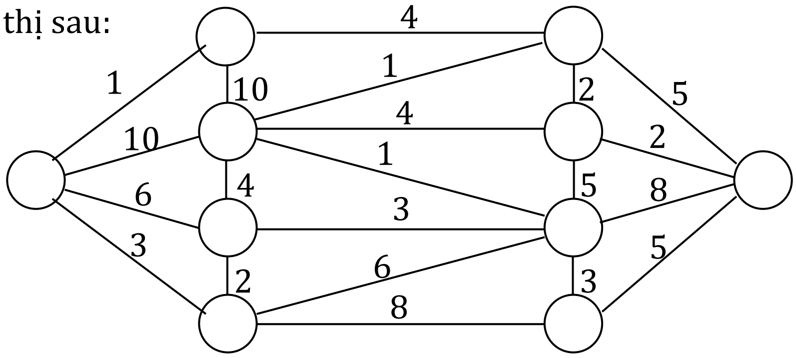
b) Chứng minh rằng P \ {v}, với v là một đỉnh bất kỳ của P, là một đồ thị Hamilton.

PHẦN 4. BÀI TOÁN TÌM HÀNH TRÌNH NGẮN NHẤT VÀ CÂY KHUNG NHỎ NHẤT

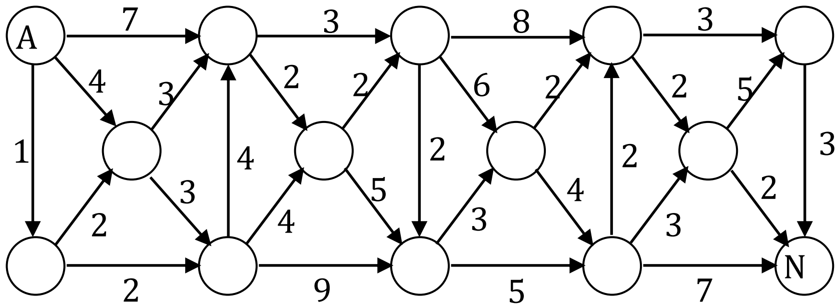
1) Dùng thuật toán Dijkstra tìm hành trình ngắn nhất từ đỉnh a đến các đỉnh khác trong đồ thị sau:



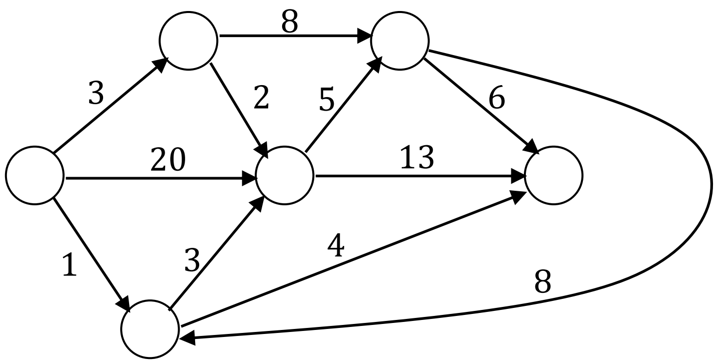
2) Dùng thuật toán Dijkstra tìm hành trình ngắn nhất từ đỉnh a đến các đỉnh khác trong đồ thị sau:



3) Cho đồ thị có trọng số như hình dưới đây. Hãy tìm hành trình ngắn nhất từ đỉnh A đến đỉnh N.



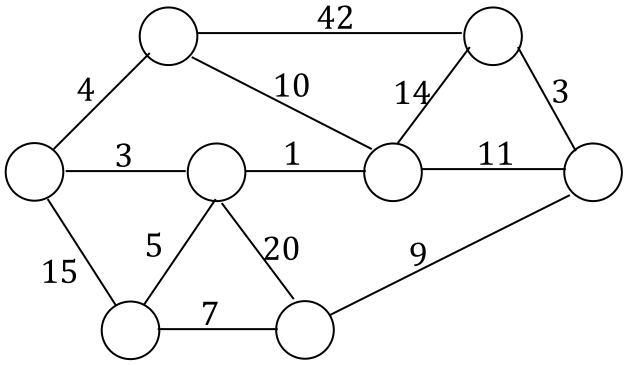
4) Tìm W* bằng cách áp dụng thuật toán Floyd vào đồ thị sau:



5) Hãy giải bài toán *người du lịch* với 6 thành phố, có số liệu cho trong ma trận trọng số sau:

$$\begin{pmatrix} \infty & 25 & 45 & 14 & 32 & 24 \\ 9 & \infty & 16 & 2 & 34 & 23 \\ 22 & 11 & \infty & 33 & 7 & 0 \\ 23 & 14 & 27 & \infty & 20 & 21 \\ 14 & 44 & 29 & 46 & \infty & 3 \\ 25 & 3 & 4 & 7 & 8 & \infty \end{pmatrix}.$$

6) Tìm cây khung nhỏ nhất của đồ thị sau theo thuật toán Kruskal và Prim.



7) Tìm cây khung nhỏ nhất bằng thuật toán Prim của đồ thị gồm các đỉnh A, B, C, D, E, F, H, I được cho bởi ma trận trọng số sau.

$$\begin{pmatrix} \infty & 16 & 15 & 23 & 19 & 18 & 32 & 20 \\ 16 & \infty & 13 & 33 & 24 & 20 & 19 & 11 \\ 15 & 13 & \infty & 13 & 29 & 21 & 20 & 19 \\ 23 & 33 & 13 & \infty & 22 & 30 & 21 & 12 \\ 19 & 24 & 29 & 22 & \infty & 34 & 23 & 21 \\ 18 & 20 & 21 & 30 & 34 & \infty & 17 & 14 \\ 32 & 19 & 20 & 21 & 23 & 17 & \infty & 18 \\ 20 & 11 & 19 & 12 & 21 & 14 & 18 & \infty \end{pmatrix}.$$

BÀI TẬP TIỂU LUẬN

BÀI 1. Khóa CDTH của Khoa Công nghệ Thông Tin, Trường CĐCN Huế có 3 lớp: A1, A2, A3. Mỗi lớp có 50 sinh viên. Cần lập thành một liên chi đoàn khóa CDTH, trong đó mỗi chi đoàn phải có tối thiểu là 2 sinh viên. Tổng sinh viên trong liên chi đoàn là 10 người. Hỏi có bao nhiêu cách lập ra liên chi đoàn nói trên.

BÀI 2. Mỗi khách hàng trong ngân hàng Đông Á sẽ được cung cấp một tài khoản. Mỗi tài khoản gồm có 2 phần: tên tài khoản và mật khẩu. Tên tài khoản và mật khẩu đều cho phép chứa các kí tự (không phân biệt hoa/thường đối với phần tên tài khoản, và có phân biệt đối với mật khẩu) và số. Độ dài của chúng là 10 kí tự. Một hacker đã viết một chương trình thử duyệt tất cả các khả năng. Giả sử thời gian để kiểm tra một mật khẩu mất 1/10 giây. Hỏi hacker đó sẽ mất bao nhiêu thời gian để có thể hack được tài khoản của một khách hàng trong trường hợp xấu nhất. Biết rằng Đông Á có 5 triệu khách hàng.

BÀI 3. Một đề thi trắc nghiệm gồm có 70 câu hỏi phần lí thuyết và 30 câu hỏi phần bài tập áp dụng. Người ta viết một chương trình máy tính cho phép hoán đổi vị trí của các câu hỏi nhưng luôn đảm bảo trong đề thi các câu hỏi lí thuyết được đặt trước các câu hỏi bài tập. Hỏi có bao nhiêu bộ đề có khả năng tạo ra.

BÀI 4. Việt nam có 64 tỉnh – thành. Giả sử các tỉnh – thành được đánh dấu theo thứ tự 1, 2, ... Giữa hai tỉnh – thành có chỉ số lẻ bất kì luôn có một đường kết nối với nhau; giữa hai tỉnh – thành có chỉ số chẵn bất kì cũng luôn có một đường kết nối với nhau. Giữa thành phố 32 và 33 có duy nhất một đường đi. Giữa hai tỉnh – thành có chỉ số chẵn và lẻ khác không có bất kì đường đi nào. Hỏi có bao nhiêu cách thực hiện một chuyến đi từ thành phố 1 đến thành phố 64.

BÀI 5. Lớp 09CDTH02 có 53 sinh viên. Trong học phần Lập trình hướng đối tượng có 40 sinh viên không thi lại và học phần Toán rời rạc có 30 sinh viên không thi lại. Nếu có 10 sinh viên thi lại cả hai môn, thì có bao nhiêu sinh viên không thi lại bất kì môn nào trong hai môn nói trên.

BÀI 6. Có bao nhiêu số trong khoảng từ 1 đến 1000, hoặc chia hết cho 2; hoặc chia hết cho 3; hoặc chia hết cho 5; hoặc chia hết cho 7.

BÀI 7. Có 5 chiếc cốc xanh, đỏ, tím, vàng và hồng. Hỏi có bao nhiêu cách sắp xếp chúng thẳng hàng sao cho:

a) Cốc màu xanh luôn gần cốc màu đỏ.
b) Cốc màu vàng luôn đứng bên trái cốc màu hồng.
c) Ba cốc xanh, đỏ và tím luôn đứng kề nhau.

BÀI 8. Một xâu kí tự được gọi là Palindrome nếu viết từ trái sang cũng như viết từ phải sang. Ví dụ abba là một Palindrome. Để duyệt tất cả các Palindrome có 10 kí tự, người ta viết một giải thuật để kiểm tra. Để kiểm tra một xâu Palindrome sẽ mất một khoảng thời gian là $10^{-6}/a$, trong đó a – là độ dài của xâu. Các kí tự của Palindrome có thể là chữ hoa/thường và các kí tự số. Hỏi:

a) Có bao nhiêu xâu Palindrome.
b) Phải mất bao nhiêu thời gian xét duyệt hết tất cả các khả năng để tìm Palindrome.

BÀI 9. Trong hệ tọa độ Decartes 3 chiều, trên mỗi mặt phẳng tọa độ XOY, YOZ, ZOX người ta đặt lần lượt 4, 5, 6 điểm. Hỏi có bao nhiêu tứ diện có thể tạo ra sao cho luôn tồn tại ít nhất một đỉnh nằm trên mỗi mặt phẳng nói trên.

BÀI 10. Chứng minh rằng nếu có 5 điểm phân biệt trong một hình vuông cạnh 2 thì có ít nhất hai điểm có khoảng cách nhỏ hơn $\sqrt{2}$.

BÀI 11. Trong một hình lập phương cạnh 1, cần có tối thiểu bao nhiêu điểm để tồn tại ít nhất 4 điểm mà thể tích của hình tạo bởi 4 điểm này nhỏ hơn 1%.

BÀI 12. Hãy chỉ ra rằng, trong một hình chữ nhật 3x4, người ta gieo vào đó 2401 điểm thì luôn tồn tại ít nhất 3 điểm sao cho diện tích của hình tạo bởi 3 điểm đó luôn nhỏ hơn 1%.

BÀI 13. Giải các hệ thức truy hồi sau đây:

a) $\begin{cases} 2x_n = 9x_{n-1} + 5x_{n-2} \\ x_0 = 1, x_1 = 3. \end{cases}$

b) $\begin{cases} x_n = 3x_{n-1} - 3x_{n-2} \\ x_0 = 1, x_1 = 2. \end{cases}$

c) $\begin{cases} x_n = 8x_{n-1} - 16x_{n-2} \\ x_0 = 2, x_1 = 1. \end{cases}$

d) $\begin{cases} x_n = 6x_{n-1} - 11x_{n-2} + 6a_{n-3} \\ x_0 = 2, x_1 = 5, x_2 = 15. \end{cases}$

BÀI 14. Giả sử tôm hùm bị đánh bắt trong một năm bằng trung bình cộng của số tôm bị đánh bắt trong hai năm trước đó. Giả sử hai năm đầu tiên đánh bắt được 1 tấn và 1.5 tấn. Hỏi Sau đó 100 năm thì sản lượng tôm đánh bắt được là bao nhiêu.

BÀI 15. Giải bài toán cái túi sau đây:

a) $\begin{cases} x_1 + 3x_2 + 2x_3 + 7x_4 \to max \\ 2x_1 + x_2 + 4x_3 + x_4 \le 4, x_i \in [0,1] \end{cases}$

b) $\begin{cases} 2x_1 + x_2 + 2x_3 + 4x_4 \to max \\ x_1 + 3x_2 + 2x_3 + x_4 \le 5, x_i \in [0,1] \end{cases}$

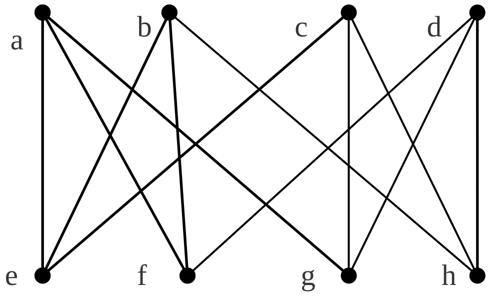
c) $\begin{cases} x_1 + 3x_2 + 2x_3 \to max \\ 2x_1 + x_2 + 2x_3 \le 4, x_i \in \mathbb{N} \end{cases}$

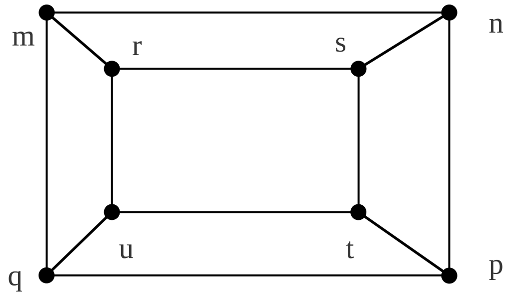
__Hướng dẫn__: Đối với câu c, ta làm hoàn toàn tương tự câu a và b. Tuy nhiên, trong bước lặp thứ nhất ta xét giá trị của x_1, trong bước lặp thứ hai, ta tiếp tục xét giá trị của x_1... và quá trình này tiếp tục đến khi không thể nào xét thêm giá trị của x_1 nữa thì ta chuyển sang xét giá trị của x_2 và tiếp tục...

BÀI 16. Giải bài toán người du lịch với ma trận chi phí sau đây:

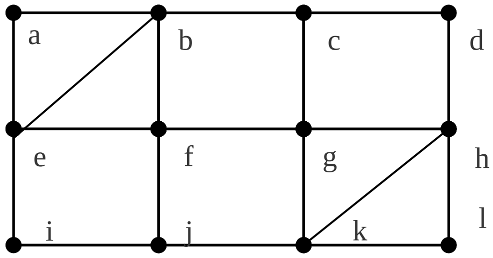
$a)\quad \begin{pmatrix} \infty & 3 & 2 & 7 \\ 4 & \infty & 6 & 11 \\ 10 & 7 & \infty & 9 \\ 3 & 8 & 7 & \infty \end{pmatrix};\qquad$ b) $\begin{pmatrix} \infty & 5 & 3 & 8 & 7 \\ 7 & \infty & 9 & 0 & 1 \\ 3 & 8 & \infty & 2 & 6 \\ 4 & 7 & 6 & \infty & 0 \\ 12 & 8 & 9 & 5 & \infty \end{pmatrix}$

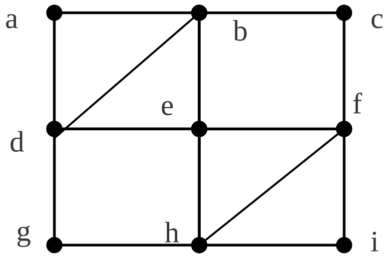
BÀI 17. Hai đồ thị sau đây có đẳng cấu không? Nếu có, hãy chỉ ra ánh xạ đẳng cấu.



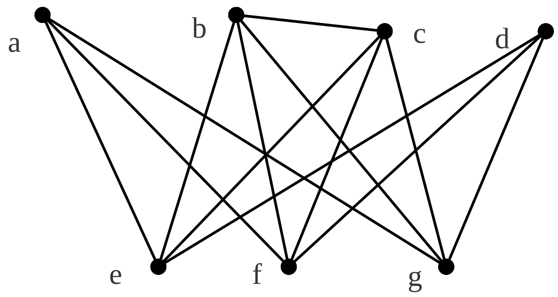


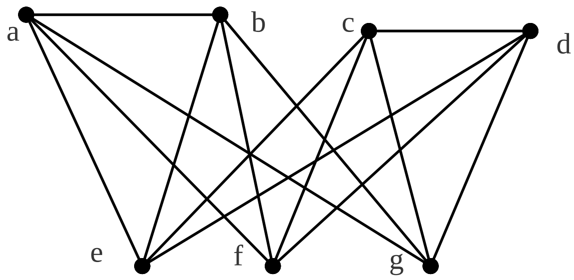
BÀI 18. Trong các đồ thị sau đây có tồn tại đường đi Euler hay chu trình Euler không? Nếu có hãy chỉ ra đường đi Euler hoặc chu trình Euler này.



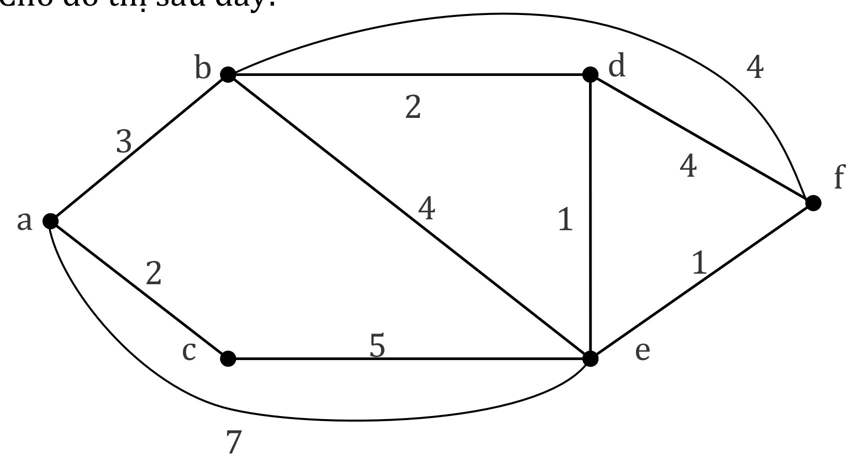


BÀI 19. Trong các đồ thị sau đây có tồn tại đường đi Hamilton hay chu trình Hamilton không? Nếu có hãy chỉ ra đường đi Hamilton hoặc chu trình Hamilton này.





BÀI 20. Cho đồ thị sau đây:



a) Tìm đường đi ngắn nhất từ $a \to f$ trong đồ thị trên bằng thuật toán Dijkstra.
b) Tìm đường đi ngắn nhất từ $a \to f$ trong đồ thị trên bằng thuật toán Floyd.
c) Tìm cây khung nhỏ nhất trong đồ thị trên bằng thuật toán Prim.
d) Tìm cây khung nhỏ nhất trong đồ thị trên bằng thuật toán Kruskal.

TÀI LIỆU THAM KHẢO

1. Kenneth H. Rosen. Discrete Mathematics and its applications. Mc-Graw Hill.

2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest. Introduction to Algorithms. MIT Press.